Carnegie Mellon University
Software Engineering Institute

Tailoring Security and Zero Trust Principles to Weapon System Environments

Christopher Alberts
Rhonda Brown
Timothy Morrow
Charles M. Wallen

September 2025

SPECIAL REPORT
CMU/SEI-2025-SR-013
DOI: 10.1184/R1/29451827

CERT® Division

https://www.sei.cmu.edu

Table of Contents

List of Tables

Abstract

Zero trust is a security model where every user, application, system, and device is untrusted by default, requiring verification and authorization for every access attempt. A key aspect of zero trust is the concept that today's infrastructures no longer have clearly defined perimeters. The movement to a zero trust philosophy changes how an organization implements its security strategy, driven by the need to manage evolving threats and technologies. Much of the available zero trust guidance focuses on applying zero trust concepts to enterprise information technology (EIT) environments. The Department of Defense (DoD) is on the path to implementing zero trust in weapon systems, which generally have different requirements than EIT systems. DoD stakeholders need guidance on how to tailor and adapt zero trust concepts to weapon system platforms. To address this need, the Software Engineering Institute (SEI) conducted a study that analyzed the applicability of foundational security and zero trust principles to weapon system environments. These principles define a framework for making security decisions and implementing security controls, enabling mission assurance through effective risk management. This report provides analysis results for nine security and zero trust principles included in the study.

1 Introduction

This report provides results of a zero trust study that members of the Software Engineering Institute (SEI), a Federally Funded Research and Development Center (FFRDC) sponsored by the Department of Defense (DoD), performed for the United States Air Force (USAF) Cyber Resiliency Office for Weapon Systems (CROWS). The objective of the study is to analyze the applicability of foundational security and zero trust principles to weapon system environments. This report provides analysis results for nine security and zero trust principles.

1.1 Zero Trust Overview

A zero trust architecture is a security model where every user, application, system, and device is untrusted by default. Each request to access computing resources must be authenticated dynamically before access is granted. National Institute of Standards and Technology (NIST) Special Publication (SP) 800-207, *Zero Trust Architecture*, defines zero trust in the following way [NIST 2020]:

> *Zero trust (ZT) is the term for an evolving set of cybersecurity paradigms that move defenses from static, network-based perimeters to focus on users, assets, and resources. A zero trust architecture (ZTA) uses zero trust principles to plan industrial and enterprise infrastructure and workflows. Zero trust assumes there is no implicit trust granted to assets or user accounts based solely on their physical or network location (i.e., local area networks versus the internet) or based on asset ownership (enterprise or personally owned). . . . Zero trust focuses on protecting resources (assets, services, workflows, network accounts, etc.), not network segments, as the network location is no longer seen as the prime component to the security posture of the resource.*

Applying zero trust principles and concepts allows an organization to shift its focus from a reactive, perimeter-focused security perspective to a proactive, data-centric strategy. This strategic shift provides several benefits, including reducing a system's attack surface, enhancing threat detection and response capabilities, improving resilience, and adapting to modern work environments while also addressing data protection and compliance requirements.

Zero trust is based on the core concept that all networks are potentially compromised, so no entity should be trusted without verification. This philosophy runs counter to organizations' traditional cybersecurity strategies and assumptions. As a result, zero trust represents a paradigm shift from the traditional cybersecurity model.

1.2 Paradigm Shift

The traditional cybersecurity model for enterprise information technology (EIT) environments employs measures and technologies to protect an organization's systems and networks from unauthorized access by establishing a secure boundary between internal and external networks. In the traditional model, the focus is on securing entry points to the organization's networks; users within the organization's security boundary are considered to be trusted. Once attackers breach perimeter security controls and gain access to an organization's infrastructure, they can traverse the infrastructure's systems and networks with relative ease [DoD 2024].

In contrast, zero trust assumes that no user or device is inherently trustworthy, requiring verification and authorization for every access attempt. A key aspect of zero trust is the concept that today's infrastructures no longer have clearly defined perimeters. The rise of cloud applications and remote work makes traditional perimeters less relevant. In addition, the zero trust philosophy recognizes that threats can originate from both inside and outside the network, making the concept of a static perimeter insufficient. This shift in philosophy introduces a significant change in implementing authentication, authorization, and security controls, and it represents a major cultural shift throughout the DoD cybersecurity ecosystem [DoD 2024]. The movement to a zero trust philosophy changes how an organization implements its security strategy, driven by the need to manage evolving threats and deploy new technologies. Zero trust provides the means to manage security risk more effectively in today's changing cyber landscape.

Zero trust principles and concepts should be applied across the entire lifecycle, from initial requirements, design, and development activities through operations and sustainment (O&S). This approach ensures that security is not an afterthought but an integral part of every phase of acquisition, development, and operations, thereby helping to minimize the risk of security breaches and improve a system's overall resilience. By integrating zero trust principles across the lifecycle, organizations can build more resilient and secure systems that are better able to withstand today's cyber threats.

Much of the zero trust guidance available from NIST and other organizations focuses on applying zero trust concepts to EIT environments. However, the DoD is on the path to implementing zero trust in weapon systems, which generally have different requirements than EIT systems. Therefore, studies need to be conducted to identify how to tailor and adapt zero trust concepts to weapon system platforms. As part of this study, the SEI analyzed how foundational cybersecurity and zero trust principles can be tailored to weapon system environments. This study started by exploring the nature of security principles and establishing their importance to managing security risks.

1.3 Importance of Security and Zero Trust Principles

A *principle* is a basic idea or concept that explains how something is supposed to work. *Security principles* codify fundamental guidelines that shape how systems, applications, and processes are designed and managed to ensure they are protected against threats and vulnerabilities. They define a basic framework for managing security risks in systems, applications, and processes. Zero trust and security are related. Zero trust defines a philosophy that enhances and strengthens traditional security practices. Zero trust augments and refines traditional security principles, such as those proposed by Saltzer and Schroeder in 1975 [Saltzer 1975], by emphasizing continuous verification and access control, even for users and devices already inside the network (considered to be trusted in traditional security models).

Security and zero trust principles are important because they provide a foundation for developing, operating, and maintaining secure systems and protecting data. These principles help to ensure that systems are protected against threats and vulnerabilities, comply with applicable laws and regulations, and are able to complete their missions. By leveraging these principles, system stakeholders can be reasonably assured that security risks are being managed and that their systems are well positioned to achieve mission success. Strategies for implementing security principles must evolve to address the dynamic nature of today's cyber landscape. The transition to zero trust likely will be iterative, requiring thoughtful change management and continuous monitoring.

1.4 SEI Zero Trust Study

Security and zero trust principles were originally designed for general-purpose computing systems, such as those found in EIT environments. As part of this study, the SEI explored how to tailor EIT-focused cybersecurity and zero trust principles to weapon system platforms that must meet stringent real-time performance requirements. The SEI team focused on accepted security and zero trust principles,[1] including the following:

- Saltzer and Schroeder's design principles for computer security [Saltzer 1975]
- additional security principles defined by Saltzer and Kaashoek [Saltzer 2009]
- DoD zero trust tenets and principles (documented in *DoD Zero Trust Reference Architecture Version 2.0*) [DISA 2022]
- DoD strategic zero trust principles (documented in *DoD Zero Trust Strategy*) [DoD 2022a]

The SEI team reviewed principles from the above sources and selected the following principles[2] to analyze in detail:

- never trust, always verify
- presume breach
- least privilege
- scrutinize explicitly
- fail-safe defaults
- complete mediation
- open design
- separation of privilege
- minimize secrets

The team made these selections after conducting a literature review of relevant publications containing principles that are generally considered to be applicable to zero trust. Table 1 provides a mapping of the principles to the source documents where they are defined.

[1] The list of cybersecurity and zero trust principles is the starting point for the study. The study may be expanded to include additional security models as appropriate. Candidate models include the Bell–LaPadula model, Biba integrity model, and Clark–Wilson integrity model.

[2] The ordering of the principles is designed to facilitate the presentation of the study's results and does not reflect their priority or level of impact.

The SEI team then collected and analyzed information about each selected principle to produce the findings documented in this report. The following format is used to present the findings for each principle:[3]

- description of the principle, including definitions from source documents
- analysis of how the principle is applied in EIT environments and considerations for tailoring the principle to weapon system environments
- questions that can be used to evaluate tradeoffs and tailoring options when applying the principle to weapon system environments

Table 1: Mapping of Principles to Source Documents

Principle	Saltzer and Schroeder	Saltzer and Kaashoek	DoD Zero Trust Reference Architecture	DoD Zero Trust Strategy
Never Trust, Always Verify			X	X
Presume Breach			X	X
Least Privilege	X	X		X
Scrutinize Explicitly			X	X
Fail-Safe Defaults	X	X		
Complete Mediation	X	X		
Open Design	X	X		
Separation of Privilege	X			
Minimize Secrets		X		

The principles the SEI team selected for analysis include both design principles from foundational research in computer security that are still relevant today as well as principles from early adopters of zero trust strategies. The selected set serves as guiding principles that align with the zero trust philosophy.

1.5 About This Report

This report provides the results of the SEI's study into the applicability of zero trust principles to weapon system environments. It provides an overview of key concepts related to zero trust and

[3] *Never trust, always verify* is a high-level principle (called a meta principle) that governs or provides a guiding framework for the other eight principles. Therefore, the section Never Trust, Always Verify includes only a description of the principle.

security risk management[4] and presents the analysis results for nine cybersecurity and zero trust principles. This report comprises the following sections:

- *Section 1: Introduction* presents a brief overview of the SEI study on tailoring zero trust principles to weapon system environments.
- *Section 2: System Concepts* provides an overview of the differences between general-purpose systems and real-time systems.
- *Section 3: Security Risk Management* provides an overview of security risk concepts, including the basic elements of security risk, types of controls, and security risk mitigation strategies.
- *Section 4: Weapon System Environment* describes four aspects of a weapon system environment that are important to understand prior to analyzing security and zero trust principles: mission context, system attributes, threat environment, and tradeoff space.
- *Sections 5-13: Analysis of Principles* provide analysis results for each principle. Sections 6 through 13 include questions to consider when tailoring security and zero trust principles to weapon system environments.[5]
  - *Section 5: Never Trust, Always Verify*
  - *Section 6: Presume Breach*
  - *Section 7: Least Privilege*
  - *Section 8: Scrutinize Explicitly*
  - *Section 9: Fail-Safe Defaults*
  - *Section 10: Complete Mediation*
  - *Section 11: Open Design*
  - *Section 12: Separation of Privilege*
  - *Section 13: Minimize Secrets*
- *Section 14: Summary and Conclusions* summarizes the report's key points and presents the high-level conclusions of the study.
- *Appendix: Zero Trust Tailoring Questions by Topic* presents the questions to consider when tailoring security and zero trust principles by security topic.

Leading practices for implementing zero trust solutions focus primarily on EIT environments. However, the DoD is committed to implementing zero trust in weapon systems, which have different requirements and mission objectives than systems in EIT environments. The next section highlights important differences between the types of systems found in each environment.

[4] The technical content in Sections 2and 3 of this report was first published in *Security Engineering Framework (SEF): Managing Security and Resilience Risks Across the Systems Lifecycle* [Alberts 2024]. The text was edited for length and clarity for inclusion in this report.

[5] Section 5 describes *Never Trust, Always Verify*, which is a meta principle for zero trust. A *meta principle* is a high-level principle that governs or provides a basis for the application of other more specific principles within a field. It acts as a guiding framework for other principles, establishing the context for making decisions and taking action. No questions are included for *Never Trust, Always Verify* due to its position as an overarching meta principle.

2 System Concepts

A *system* is defined as “an aggregation of system elements and enabling system elements to achieve a given purpose or provide a needed capability” [DoD 2022b]. The differences between EIT systems and weapon systems are important because of their distinct functions, performance requirements, and security needs. EIT environments incorporate general-purpose systems that are designed for information processing, communication, and business operations. In contrast, weapon systems are typically real-time systems that are engineered for their destructive capabilities and combat operations. This section explores key differences between general-purpose and real-time systems. Understanding these differences is important when tailoring security and zero trust principles developed for EIT environments to weapon systems.

2.1 General-Purpose Systems

A *general-purpose system* is designed to be user friendly, run a variety of applications, and support multiple users and devices. As a result, it often prioritizes functionality and diversity over speed and reliability. A general-purpose system is thus more flexible and versatile than a real-time system, but it does not guarantee a specific level of performance or responsiveness. It is common for a general-purpose system’s user to experience delays, errors, or crashes, depending on the workload and processing resources available. Examples of general-purpose systems include mainframe computers, servers, laptops, desktop computers, smartphones, and tablets.

2.2 Real-Time Systems

A *real-time system* is one in which computation must be performed during the actual time that an external process occurs, allowing computational results to respond to those external processes [DAU 2024]. This type of system enables real-time control over hardware resources by providing deterministic behavior and predictable response times. Because it can manage concurrent tasks, a real-time system ensures consistent operation even under extreme loads and varying conditions. Examples of real-time systems include industrial control systems, automobile-engine fuel injection systems, medical imaging systems, command-and-control systems, and weapon systems.

2.3 Managing Tradeoffs in Real-Time and General-Purpose Systems

Security risks are often managed differently in real-time and general-purpose systems. One key difference is the tradeoffs among quality attributes for each type of system. Quality attributes are the functional and nonfunctional requirements that are used to evaluate system performance [Hilburn 2023]. Examples of quality attributes include performance, security, reliability, interoperability, usability, portability, maintainability, and scalability. Depending on the system being designed and developed, some quality attributes are more important than others. Quality attributes are typically prioritized differently in real-time and general-purpose systems. A key tradeoff that must be considered is performance versus security requirements.

For example, consider how the performance and security tradeoff is addressed in the two types of systems. A security requirement that introduces latency into a system's processing could introduce an unacceptable performance risk in a real-time system. In a military operation, a system's response time can be the difference between mission success and failure. As a result, an ordnance could miss its target or information might be received a few seconds too late by a command-and-control system. In both cases, system processing delays could result in the failure of an operational mission.

As illustrated in the example, engineers and developers might decide to accept a security risk to meet a real-time system's performance requirements. However, that tradeoff would likely be viewed very differently for a general-purpose system, such as an enterprise accounting system. The performance requirements of an accounting system are very different from those of a weapon system or a command-and-control system. Accounting processes can tolerate a degree of processing delays and system crashes that are unacceptable in a real-time system. The latency introduced by the security requirement likely will not affect the accounting system's performance in a meaningful way.

Security practices, including zero trust practices, can be applied to both real-time and general-purpose systems. However, they will be implemented quite differently in those systems based on disparities in the systems' risks and tradeoffs. Weapon system stakeholders need to tailor zero trust solutions and design choices based on a systematic analysis of the tradeoffs among the system's requirements, risks, and mission objectives.

3 Security Risk Management

Security and zero trust principles define a framework for managing a system's security risks, providing the context for making security decisions and implementing appropriate controls. This section presents basic security risk management concepts. It also provides a mapping of the security and zero trust principles to the risk mitigation strategies that they support, ensuring that the principles included in this study have sufficient breadth from a security risk perspective.

Effective security risk management requires establishing and maintaining protective measures that enable an organization to achieve its mission despite the risks posed by threats to its systems. Protective measures may involve combining protection, detection, response, and recovery to provide the basis for an organization's risk management approach [NIST 2021]. From a security perspective, risk management comprises the activities required to manage security risks to operations, systems, and individuals within and across organizations. Security risk management is an important organizational practice that provides a basis for selecting and applying security controls to systems.

3.1 Security Controls

Security controls are the safeguards or countermeasures prescribed for an information system or organization to protect the confidentiality, integrity, and availability (CIA) of the system and its information [NIST/DOC 2020]. There are three basic types of security controls:

- *Technical controls*—safeguards or countermeasures that are primarily implemented and executed through mechanisms contained in system components [NIST/DOC 2012] (Examples of technical controls include firewalls, intrusion detection systems [IDSs], encryption, and identification and authentication mechanisms.)
- *Physical controls*—mechanisms that deny unauthorized access to facilities, equipment, and resources and protect personnel and property from damage or harm (Examples of physical controls are card readers, cameras, motion sensors, intruder alarms, equipment inventories, surge protectors, and fire protection.)
- *Administrative controls*—policies, procedures, or guidelines that define personnel and organizational practices in accordance with the organization's security goals (Examples of administrative controls are security training and awareness programs, password management policies, and incident response planning.)

From a lifecycle perspective, all three types of security controls must be implemented. During system design and development, the emphasis is on implementing technical controls at the system level. The security and zero trust principles addressed in this study provide the framework for making security decisions and implementing technical security controls, which enable mission assurance through effective risk management.

3.2 Security/Resilience Risk Mitigation Strategies

Security risk is a measure of (1) the likelihood that a threat will exploit a vulnerability to produce an adverse consequence or loss and (2) the magnitude of the loss [Alberts 2014]. Options for handling security risks are selected during risk analysis. High-priority security risks are generally mitigated to reduce or contain the risks. Broad-based mitigation plans for security risks incorporate the following basic strategies:[6]

- *Protect*. Reduce the vulnerability to threats and minimize any consequences that might occur.
- *Detect*. Identify the occurrence of a security/resilience threat (i.e., cyber attack).
- *Respond*. Take action to counteract a detected threat (i.e., cyber attack) and minimize consequences, losses, and damages.
- *Recover*. Restore access to and functionality of a system (or systems) after a risk's consequences, losses, and damages are realized.
- *Adapt*. Enable a sustained capability to accommodate changes in a system's risk environment, including changes to threats, vulnerabilities, missions, and technologies.

A system's risk mitigation plan will specify a set of security controls. Once a mitigation plan is developed, documented, and approved, stakeholders must obtain resources and then implement and manage the plan. An effective risk mitigation plan for a system typically addresses all five mitigation strategies to provide a layered risk mitigation approach that enhances the system's overall security and resilience.

Table 2 provides a mapping of the security and zero trust principles to the mitigation strategies they support. The principles provide the context for making security decisions and implementing security controls that reduce the vulnerabilities and security risks in systems across the lifecycle. It is thus important to ensure that the principles address all five of the strategies needed to mitigate security risks effectively (i.e., protect, detect, respond. recover, and adapt). A key aspect of the study's design is to ensure that the principles have sufficient breadth from a security risk perspective, and Table 2 shows that the principles address all five mitigation strategies.

Table 2 also shows relationships among the principles when viewed from a risk mitigation perspective. For example, *detect* maps to four of the nine principles. The relationships among these four principles provide an opportunity to leverage security resources, practices, and controls for detecting the presence of security threats and thus increase the potential for mission success.

[6] The mitigation strategies presented in this report are a composite of strategies defined for security, resilience, and survivability. Cybersecurity mitigation strategies are sourced from *Framework for Improving Critical Infrastructure Cybersecurity, Version 1.1* [NIST 2018]. Strategies for mitigating resilience risks for a system are sourced from *Developing Cyber-Resilient Systems: A Systems Security Engineering Approach* [NIST 2021]. Finally, strategies that contribute to the survivability of a system's capabilities are sourced from the *Cyber Survivability Endorsement (CSE) Implementation Guide, Version 3.0* [DoD 2022c].

Table 2: Mapping of Principles to Mitigation Strategies

Principles	Protect	Detect	Respond	Recover	Adapt
Never Trust, Always Verify	X	X	X	X	X
Presume Breach	X	X	X	X	
Least Privilege	X				
Scrutinize Explicitly		X	X		X
Fail-Safe Defaults	X				
Complete Mediation	X	X	X		X
Open Design	X				
Separation of Privilege	X				
Minimize Secrets	X				

Finally, the significance of each principle is best understood when it is viewed within the context where it is applied. This view is an important part of the tailoring process. EIT and weapon system environments are very different. The next section describes key aspects of the weapon system environment that must be characterized before tailoring activities can begin.

4 Weapon System Environment

The first step when tailoring security and zero trust principles to a weapon system platform is to analyze the environment where the weapon system will operate. Four aspects of a weapon system's environment are particularly important to understand: mission context, system attributes, threat environment, and tradeoff space.

4.1 Mission Context

The *mission context* for a weapon system refers to the purpose, goals, and operational environment in which a system is designed, developed, deployed, operated, and maintained. The mission context provides critical variables that can influence mission outcomes and decisions. It should also include enough information to address the investigative questions about a weapon system and its objectives [DoD 2023a]. The following questions can be used to elicit information about a weapon system's mission context:[7]

- What mission does the weapon system support?
- What key assets and capabilities are essential for mission success?
- What are the dependencies between these assets and capabilities? How do they affect each other?
- How does the weapon system support the mission? What are the weapon system's critical interfaces?
- What risks could affect the potential for mission success? How might these risks degrade or disrupt the mission?
- Does the weapon system have separate security requirements for mission and maintenance activities? If yes, how is this difference addressed in practice?

4.2 System Attributes

A *system attribute* is a quality or feature that defines an important characteristic of a weapon system. Unlike systems operating in EIT environments, legacy and newly acquired weapon systems have unique constraints that can limit the extent to which zero trust capabilities can be implemented [DoD 2023b]. A relatively small set of system attributes can provide stakeholders with the information they need to evaluate the weapon system's ability to accommodate zero trust capabilities. The following five attributes were identified by the DoD: dynamic configurability;

[7] A weapon system can support multiple missions. The questions for mission context should be reviewed and answered for each high-priority mission that a weapon system supports.

design/retrofit flexibility; size, weight, and power (SWaP); latency tolerance; and information technology (IT)/operational technology (OT)-centricity [DoD 2023b].[8]

Answering the following questions can help evaluate the key attributes for a weapon system:[9]

- ***Dynamic Configurability***: How much flexibility does the weapon system have to accommodate dynamic, system-level changes to its security configuration baseline?

  Answers

  - constrained (low flexibility)
  - partially constrained (moderate flexibility)
  - unconstrained (high flexibility)

- ***Design/Retrofit Flexibility***: How flexible is the architecture for the modifications and retrofitting required to implement zero trust?

  Answers

  - rigid (low architecture flexibility)
  - moderate (moderate architecture flexibility)
  - agile (high architecture flexibility)

- ***SWaP***: How much will the weapon system's size, weight, and power requirements constrain or limit the ability to implement zero trust capabilities in the weapon system?

  Answers

  - constrained (minimal SWaP changes tolerated)
  - partially constrained (low to moderate SWaP changes tolerated)
  - unconstrained (SWaP changes not an issue)

- ***Latency Tolerance***: How much latency from implementing zero trust capabilities can the weapon system tolerate?

  Answers

  - none (no ability to tolerate latency)
  - low (low degree of latency tolerance)
  - high (high degree of latency tolerance)

- ***IT/OT-Centricity***: What types of software components are included in the weapon system?

  Answers

  - OT
  - hybrid
  - IT

8 The analysis results for the five zero-trust system attributes are sourced from *Zero Trust: Analysis of the Applicability to Weapon Systems and Defense Critical Infrastructure, Version 1.0* [DoD 2023b]. Read this document for an in-depth treatment of the attributes and examples of how to apply them.

9 The answers provided for each question are defined in *Zero Trust: Analysis of the Applicability to Weapon Systems and Defense Critical Infrastructure, Version 1.0* [DoD 2023b]. The text in paratheses provides additional context relative to the wording of each choice.

4.3 Threat Environment

The *threat environment* for a system includes the range of risks that can affect a system's security and lead to adverse mission impacts. Security analysts must consider the internal and external threats and the context in which these threats operate when assessing the risks to a system. A security risk assessment requires analysts to identify the actors that might target the system, analyze the system's vulnerabilities, and determine the impact on the mission that the system supports. Answering the following questions can help characterize the threat environment for a weapon system:

- Which threat actors might target the weapon system?
- How motivated are those threat actors?
- How could threat actors attack the weapon system?
- How vulnerable is the weapon system to those attacks?
- What would be the mission impact(s) of these attacks if successful?
- Which security controls are currently in place to mitigate the risk from these attacks?

4.4 Tradeoff Space

In the context of systems and software engineering, the *tradeoff space* refers to the range of possible solutions or design choices that require engineers to strike a balance among competing requirements or objectives. As a result, engineers must investigate different combinations of features, performance characteristics, and constraints to make informed decisions. For example, when making decisions about selecting and implementing security controls for a system, engineers need to balance security requirements with performance and interoperability requirements as well as program's cost and schedule objectives. The following questions can provide insight into the tradeoff space for a system:

- What are the performance requirements for the system?
- What are the security, resilience, and survivability requirements for the system?
- What are the interoperability requirements for the system?
- How could program constraints (e.g., budget, schedule) affect the program's ability to mitigate the weapon system's security risks?
- What other program requirements, constraints, and objectives could affect the program's ability to mitigate the weapon system's security risks?

The questions presented in this section are important to answer for understanding a weapon system's environment. The answers to these questions will help answer the security and zero trust questions provided for each principle included in the remainder of this report.

5 Never Trust, Always Verify

Never trust, always verify is a meta principle[10] of zero trust. According to this principle, no user, device, or network location is inherently trusted. Every access request must be verified and authenticated before access to computing resources is granted, regardless of where the request originates. Resource authentication and authorization are dynamic and strictly enforced before access is allowed. Following this principle requires a continuous cycle of obtaining access, scanning and assessing threats, adapting, and continually reevaluating trust in ongoing communication [NIST 2020].

Key definitions of *never trust, always verify* from source documents include the following:

- "Treat every user, device, and application as untrusted and unauthenticated. Authenticate and explicitly authorize to the least privilege using dynamic security policies" [DoD 2022a].
- "Deny access by default. Every device, user, application/workload, and data flow are authenticated and explicitly authorized using least privilege, multiple attributes, and dynamic cybersecurity policies" [DISA 2022].

Never trust, always verify establishes a common foundation for the other security and zero trust principles included in the study. It defines high-level concepts that are used to organize and interpret the remaining eight principles. Refer to the analysis of each of the remaining principles in this report for detailed information about that principle and guidance for tailoring it to weapon system environments.

[10] A *meta principle* is a high-level principle that governs or provides a basis for the application of other more specific principles within a field. It acts as a guiding framework for other principles, establishing the context for making decisions and taking action.

6 Presume Breach

The zero trust principle of *presume breach* means that an organization should assume that its networks have already been compromised. As a result, no user, application, system, or device should be trusted by default, which requires continuous verification and validation of every access request. Key definitions of *presume breach* from source documents include the following:

- "Consciously operate and defend resources with the assumption that an adversary already has presence within the environment. Deny by default and heavily scrutinize all users, devices, data flows, and requests for access. Log, inspect, and continuously monitor all configuration changes, resource accesses, and network traffic for suspicious activity" [NSA 2021].
- "There are hundreds of thousands of attempted cybersecurity attacks against DoD environments every day. Consciously operate and defend resources with the assumption that an adversary has presence within your environment. Enhanced scrutiny of access and authorization decisions to improve response outcomes" [DISA 2022].
- "Limit the 'blast radius'—the extent and reach of potential damage incurred by a breach—by segmenting access, reducing the attack surface, and monitoring risks in real-time within DoD's risk tolerance levels and thresholds" [DoD 2022a].

Traditional security strategies focus on addressing perimeter defenses in an effort to prevent security incidents. *Presume breach* expands that focus to include internal protections. Organizations should assume that there is a malicious presence inside their environment and implement security controls to minimize the impact. This mindset requires minimizing the impact and spread of breaches by keeping the consequences of an attack compartmentalized. *Presume breach* is often coupled with the principle of *least privilege*, meaning that users and non-person entities should be granted only the minimum level of access needed to perform their job functions.

6.1 Analysis

In EIT environments, every user, device, and request must be verified before granting access to any data or system, regardless of its location within the network. The principle of *presume breach* requires thorough verification of the identity and access rights for each user, application, system, and device attempting to access a resource. [11] Verification is addressed without making assumptions about the trustworthiness of the user, application, system, or device that is requesting access. Resources must be accessed in a secure manner using multiple factors to establish confidence levels and confirm legitimacy before access is granted to those resources. All access requests are continuously monitored and analyzed in near real time for both user and device behaviors.

Addressing the principle of *presume breach* also requires minimizing the impact and spread of breaches by keeping consequences of an attack compartmentalized. A variety of controls are

[11] The phrase *access a resource* refers to any attempt by a user, application, system, or device to gain entry to data, systems, or services within a network. In a zero trust architecture, all access requests are rigorously verified and authenticated before being granted.

implemented in EIT environments to manage security risks, including architecture, authentication, encryption, monitoring, response, and recovery controls.

The impact and spread of breaches can be minimized by dividing the system's architecture into small, isolated segments. Segmenting the architecture restricts the lateral movement of attackers within the network. Strong authentication controls can be implemented to verify every user and device that attempts to access data and services in the EIT environment. No user, device, or system is automatically trusted, even if it is already inside the network perimeter. Data at rest and in transit can and should be encrypted to protect data confidentiality.

Monitoring plays a key role in EIT environments. Access requests should be monitored continuously to detect potential threats early. User identities, device health, and access requests should be monitored continuously to detect potential threats. In addition, systems should be monitored for security threats during operations. Threat information provides a foundation for establishing situational awareness for a system. By incorporating threat information with internal system data, stakeholders can develop a holistic view of a system's security posture, enabling them to proactively defend the system against threats and respond effectively to incidents.

Security events and incidents (i.e., disruptions) are occurrences that actually or potentially jeopardize the confidentiality, integrity, or availability (CIA) attributes of an operational system. A response plan for a system outlines actions that will be taken in response to a specific event or incident. Its purpose is to provide a clear framework for identifying, assessing, and mitigating potential risks, ensuring a coordinated and effective response. A system recovery plan provides a coordinated strategy for restoring a system and its data after a disruption. Effective recovery plans include strategies and practices that minimize the impact of disruptions and restore normal system functionality as quickly as possible.

Highly motivated and well-funded threat actors (such as nation states) have the potential to direct sophisticated cyber attacks on weapon systems. These threat actors may launch cyber attacks directly on weapon systems or indirectly via support systems (e.g., maintenance systems) or via supply chain dependencies. It is thus safe to assume that *presume breach* applies to weapon systems.

Weapon system stakeholders need to manage the tradeoffs between performance requirements and security controls. These tradeoffs are managed differently in EIT and weapon system environments. As a result, weapon system stakeholders might make different choices regarding security controls due to the different trade spaces. A model of the weapon system's architecture facilitates the analysis of cyber risks and can help analysts assess tradeoffs between performance requirements and security controls.

Many weapon systems, such as aircraft, are built on commercial platforms. A weapon system normally inherits the architecture of the commercial platform upon which it is built. The potential for dividing the commercial platform's architecture into small, isolated segments might be limited due to cost and support considerations. Similarly, the architecture of a legacy weapon system might not be easily modified to address the zero trust principle of *presume breach*. Adding mission subsystems to a commercial platform and upgrading mission subsystems on a legacy platform provide some opportunities for addressing zero trust principles only at the local level (i.e., in the subsystem architecture, not in the overall system architecture).

The performance versus security tradeoffs of implementing authentication, encryption, monitoring, response, and recovery controls in weapon system environments will differ from those in EIT environments. For example, controls that introduce latency into a weapon system's processing could introduce unacceptable mission risks. Weapon system stakeholders might need to relax some zero trust controls and accept the resulting security risks to meet the system's performance requirements.

6.2 Questions

Is dividing the system's architecture into small, isolated segments (to restrict lateral movement of attackers within the network) practical for the weapon system?

- Will a highly segmented architecture have an impact on operational performance and mission success?
- Is a highly segmented architecture practical for weapon systems that are built on commercial or legacy platforms?

Should encryption be implemented in the weapon system to protect data at rest and in transit?

- Will latency introduced by encryption capabilities adversely affect mission performance requirements?

Are there conditions where a deny-access state may not be a safe default state?

- Are there circumstances where an operator absolutely must be able to take control of a weapon system?
- Does design guidance describe when deny by default could be an unsafe condition?

Can every access request, regardless of location or source, be authenticated and authorized using multiple attributes (e.g., multi-factor authentication, device health checks, user context)?

- Will latency introduced by authentication capabilities adversely affect mission performance requirements?

Is it feasible to perform a device posture check to verify a device's current security status before granting access to a weapon system's resources?

- Can a device's security posture be assessed continually to ensure it meets policies for accessing sensitive data?
- Can devices be checked during military operations to ensure that they are up to date with patches and have appropriate security software?

Can the weapon system be monitored for security threats during mission execution?

- Is data collected, analyzed, and communicated to provide adequate situational awareness of the weapon system's threat environment?
- Is it feasible to implement real-time security analytics to detect anomalies and potential threats during mission execution?

Have response and recovery plans been developed for the weapon system?

- Has an incident response plan for managing security events and incidents been developed and tested for the weapon system?
- Has a recovery plan for restoring a system and its data after a disruption been developed and tested for the weapon system?

Are automation and analytics implemented to manage large volumes of data for collecting, analyzing, and correlating security data for the weapon system and its subsystems/components?

- Will the processing needed to collect, analyze, and manage security data adversely affect mission performance requirements (e.g., by introducing latency)?
- Can dynamic policy adjustments for the weapon system and its subsystems/components introduce interoperability or performance risks to the mission?

What workarounds or changes related to the *presume breach* principle might be needed in the weapon system's operational mission environment?

7 Least Privilege

The principle of *least privilege* states that users, applications, systems, and devices should be able to access only the minimum resources and permissions needed to perform their assigned tasks. *Least privilege* significantly reduces an organization's attack surface by restricting access to an organization's IT resources, which minimizes the potential damage resulting from a security breach. Key definitions of *least privilege* from source documents include the following:

- "Every program and every user of the system should operate using the least set of privileges necessary to complete the job. Primarily, this principle limits the damage that can result from an accident or error. It also reduces the number of potential interactions among privileged programs to the minimum for correct operation, so that unintentional, unwanted, or improper uses of privilege are less likely to occur. . . . The military security rule of need-to-know is an example of this principle" [Saltzer 1975].
- The concept of least privileged access refers to eliminating "the idea of trusted or untrusted networks, devices, personas, or processes, and shifts to multi-attribute-based confidence levels that enable authentication and authorization policies" [DoD 2022a].

This principle is related to the *separation of privilege* principle, where no single entity possesses all the necessary permissions to compromise the security of a system or access critical resources.

7.1 Analysis

In an EIT environment, access permissions for users are generally based on organizational roles and responsibilities, which tend to be relatively static over time. Changes to access permissions for users can be planned and managed. For example, users' access permissions can be updated based on changes to their roles and responsibilities. Access permissions for non-person entities (e.g., applications, systems, devices) are based on organizational policy. Adding or removing access for applications, systems, and devices also can generally be planned and managed in an EIT environment.

For a zero trust architecture (as implemented in an EIT environment), access is evaluated and granted on a per-session basis. Each access request requires authentication and authorization. Privileges are granted long enough for a user or non-person entity to perform a specific task. In addition, access to an organization's resources is not limited by geographic location. Zero trust requires that every user, application, system, and device attempting to access a resource is authenticated before gaining access regardless of location.

Changes to users' access permissions are typically planned in EIT environments. When users assume new roles and responsibilities, their access permissions are adjusted accordingly. These changes can be planned and normally do not require real-time adjustments. In contrast, weapon systems are deployed in unpredictable and highly contested environments, where real-time adjustments to users' access permissions might be needed. Weapon system stakeholders must determine the extent to which access requirements or security status might change dynamically during mission execution and be able to respond accordingly.

Weapon system stakeholders must assess zero trust requirements and tradeoffs related to the principle of *least privilege*. For example, it might not be feasible to restrict access privileges on a per-session basis. This limitation could introduce issues (e.g., latency) that could affect mission execution (and ultimately mission success). A thorough risk analysis will help stakeholders balance zero trust and mission requirements by examining the associated risks and tradeoffs.

7.2 Questions

Can roles and requirements for users or non-person entities (e.g., applications, systems, devices) change dynamically during mission execution (i.e., when the weapon system is supporting an operational mission)?

Can access permissions to the weapon system's capabilities change in real time?

- Are access permissions considered for both users and non-person entities (e.g., applications, systems, devices)?

Are there instances when users or non-person entities (e.g., applications, systems, devices) will unexpectedly need elevated privileges to support the weapon system's mission?

Are there instances where access privileges need to persist beyond a session (i.e., not limited by time and scope)?

Which applications, systems, and devices can access the weapon system during mission execution?

- How are these dependencies managed?

Which applications, systems, and devices can access the weapon system during system maintenance activities?

- How are these dependencies managed?

What workarounds or changes related to the *least privilege* principle might be needed in the weapon system's operational mission environment?

8 Scrutinize Explicitly

The zero trust principle of *scrutinize explicitly* involves verifying and authenticating access requests based on the available data for each user, application, system, and device. The data used for verification and authentication typically includes user identity, device health, location, and data classification. Key definitions of *scrutinize explicitly* from source documents include the following:

- "All resources are consistently accessed in a secure manner using multiple attributes (dynamic and static) to derive confidence levels for contextual access to resources. Access to resources is conditional and access can dynamically change based on action and confidence levels resulting from those actions" [DISA 2022].
- "All events within our information environment must be continuously monitored, collected, stored, and analyzed based on risk profiles and generated in near-real time for both user and device behaviors" [DoD 2022a].

Scrutinize explicitly incorporates the following zero trust practices:

- *no trust by default*—Every access request is analyzed. Analysis of access requests is not limited to those that pose the greatest risk.
- *multi-factor authentication*—Verification of identity requires two or more factors to gain access to a resource. This practice helps prevent unauthorized access to accounts and applications.
- *dynamic access control*—Access decisions are made based on real-time verification. Dynamic analysis enables adjustments to access permissions based on changing circumstances and conditions.
- *policy enforcement*—Explicit policies are needed to successfully implement and enforce dynamic access control.

S*crutinize explicitly* requires that resources are consistently accessed in a secure manner using multiple attributes. By verifying and authenticating every access request, the risk of unauthorized access and data breaches is reduced.

8.1 Analysis

In a zero trust architecture, the principle of *scrutinize explicitly* requires thorough verification of the identity and access rights for each user, application, system, and device attempting to access a resource.[12] Verification is addressed without making assumptions about the trustworthiness of the user, application, system, or device that is requesting access. Resources must be accessed in a secure manner using multiple factors to establish confidence levels and confirm legitimacy before

[12] The phrase *access a resource* refers to any attempt by a user, application, system, or device to gain entry to data, systems, or services within a network. In a zero trust architecture, all access requests are rigorously verified and authenticated before being granted.

access is granted to those resources. All access requests are continuously monitored and analyzed in near real time for both user and device behaviors.

Resource authentication and authorization are dynamic and strictly enforced before access is allowed. This practice requires a continuous cycle of obtaining access, scanning and assessing threats, updating access policies and procedures accordingly, and reevaluating trust continually. The following security concepts are addressed by *scrutinize explicitly*: user and asset inventories, identity verification, device posture checks, continuous monitoring, policy enforcement, and automation and analytics.

- A *user and asset inventory* refers to a comprehensive list of users, applications, systems, and devices within an organization. It provides the foundation for managing identities in a zero trust environment. Establishing a user and asset inventory ensures that all users, applications, systems, and devices with access to critical resources are vetted and registered.
- *Identity verification* requires every access request, regardless of location or source, to be authenticated and authorized using multiple attributes. A zero trust architecture employs *dynamic access control* for authentication and authorization. Dynamic access control continually evaluates and adjusts access to resources based on real-time factors like user location, device health, current activity, and other contextual data. As a result, access permissions are not static (i.e., relying on predefined user credentials and user roles) but are dynamically granted and revoked based on multiple real-time factors. [13]
- *Device posture checks* (i.e., *health checks*) are used to verify a device's current security status before granting access to an organization's EIT resources. This means that a device's security posture is assessed continually to ensure it meets the organization's policies for accessing sensitive data.
- *Continuous monitoring* is an important aspect of applying *scrutinize explicitly* in EIT environments. *Continuous monitoring* refers to the ongoing, real-time observation and analysis of user activity, device behavior, and network traffic to continuously verify access rights and detect potential security threats. Real-time monitoring requires employing security analytics to detect anomalies and potential threats.
- *Policy enforcement* refers to the continuous monitoring and verification process that occurs at all access points. Every user, application, system, and device attempting to access a resource is continually analyzed and authenticated before access to sensitive data or systems is granted. A *zero trust policy* is the set of rules that govern how an organization implements a zero trust architecture and corresponding security model. The policy defines how to verify access requests and grant or deny access to corporate resources. A *policy enforcement point (PEP)* is a central component in a zero trust architecture that actively enforces access control policies by checking user permissions against defined rules and deciding whether to grant or deny access to a resource. It acts as a gatekeeper to protect digital assets within a network.

[13] Static access control relies on a set of predefined user identification parameters, such as username and password as well as user roles. In contrast, dynamic access control adapts access rights in real time based on multiple factors, such as user location, time of access, device health, and the specific resource being accessed. As a result, dynamic access control provides for more flexible and context-aware management of access permissions.

- *Automation and data analytics* can play an important role when implementing zero trust policy enforcement. For example, data analytics can be employed to monitor network traffic patterns to identify anomalous behavior or suspicious activity; analyze user actions to detect unusual activity; and correlate network data with external threat data to proactively identify risks. These actions enable more dynamic and context-aware policy adjustments based on real-time data analysis. While automation and data analytics are not required to enforce zero trust policy, they are important for optimizing an organization's zero trust architecture.

For weapon system platforms, stakeholders must assess zero trust requirements and tradeoffs related to the principle of *scrutinize explicitly*, particularly in relation to user and asset inventories, identity verification, device posture checks, continuous monitoring, policy enforcement, and automation and analytics. The practices needed to implement this principle could introduce risks that affect mission execution. For example, the technologies required to implement continuous monitoring and policy enforcement could affect a weapon system's performance by consuming system resources and introducing latency. A thorough risk analysis will help stakeholders balance zero trust and mission requirements by examining the associated risks and tradeoffs.

8.2 Questions

Is it feasible to establish a comprehensive list of users, applications, systems, and devices for the mission that the weapon system supports?

- Is the mission environment too dynamic to create a comprehensive user and asset inventory?
- Could interoperability issues arise if users, applications, systems, and devices not in the inventory are assigned to participate in the mission?

Can every access request, regardless of location or source, be authenticated and authorized using multiple attributes (e.g., multi-factor authentication, device health checks, user context)?

- Will latency introduced by authentication capabilities adversely affect mission performance requirements?

Is it feasible to perform a device posture check to verify a device's current security status before granting access to a weapon system's resources?

- Can a device's security posture be assessed continually to ensure it meets policies for accessing sensitive data?
- Can devices be checked during military operations to ensure that they are up to date with patches and have appropriate security software?

Can continuous monitoring be implemented in the weapon system?

- Can user activity be monitored in real time for suspicious behavior?
- Is it feasible to implement real-time security analytics to detect anomalies and potential threats during mission execution?

Is a zero trust policy developed for the weapon system and its subsystems/components as appropriate?

- Does the zero trust policy for the weapon system and its subsystems/components align with mission requirements?
- Could interoperability issues arise if the zero trust policy for the weapon system is not aligned with mission requirements?

Are automation and analytics implemented to manage large volumes of data for collecting, analyzing, and correlating security data for the weapon system and its subsystems/components?

- Will the processing needed to collect, analyze, and manage security data adversely affect mission performance requirements (e.g., by introducing latency)?
- Can dynamic policy adjustments for the weapon system and its subsystems/components introduce interoperability or performance risks to the mission?

What workarounds or changes related to the *scrutinize explicitly* principle might be needed in the weapon system's operational mission environment?

9 Fail-Safe Defaults

The *fail-safe defaults* principle denies access to resources or information by default unless permission is granted explicitly. This means that a system should always restrict access unless it is actively authorized, minimizing the risk of unauthorized access or security breaches.

The following is a key definition of *fail-safe defaults* from a source document:

- "Base access decisions on permission rather than exclusion. . . . The default situation is lack of access, and the protection scheme identifies conditions under which access is permitted. . . . A design or implementation mistake in a mechanism that gives explicit permission tends to fail by refusing permission, a safe situation, since it will be quickly detected. On the other hand, a design or implementation mistake in a mechanism that explicitly excludes access tends to fail by allowing access, a failure which may go unnoticed in normal use" [Saltzer 1975].

The principle of *fail-safe defaults* defines the default access state for users, applications, systems, and devices. When a new user account is created, the user will not have access to any applications, systems, devices, or data in the enterprise. Access to those resources must be provided by an appropriate administrator. Likewise, access to any code repository must be granted by an administrator; there are no exceptions for roles, responsibilities, or positions.

It is important to note that the *fail-safe defaults* principle is different than a fail-safe state. A *fail-safe state* refers to the condition a system automatically enters when a failure occurs, preventing adverse consequences. The principle of *fail-safe defaults*, as outlined by Saltzer and Schroeder, is a design philosophy that focuses on access control and permissions [Saltzer 1975]. It focuses on the initial configuration of a system, denying access unless it is granted explicitly. A fail-safe state is a general concept about system behavior in the event of failure; it does not specifically address access control and a system's initial configuration.

9.1 Analysis

The focus of the *fail-safe defaults* principle is denying access to data and resources by a user or non-person entity by default unless it is explicitly granted. The last part of this statement results in this principle being closely aligned with the principle of *least privilege*. The principle of *least privilege* limits access to resources based on what is necessary to perform a task. The *fail-safe defaults* principle restricts how privileges are initialized when data and resources are created.

In an EIT environment, access permissions for users are generally based on organizational roles and responsibilities. A user is granted access permissions based on the *least privilege* principle for their role. If the user does not have a need to access an object or resource, then—based on *fail-safe defaults*—the user is denied access. For efficiency, IT personnel use templates to provision new users and assign role-based access privileges. A review of IT templates should be performed that focuses on both the *fail-safe defaults* and *least privilege* principles.

Another context in which the *fail-safe defaults* principle applies is the software supply chain. “A software supply chain is the network of stakeholders that contribute to the content of a software product or that have the opportunity to modify its content” [Dorofee 2003]. A software product comprises many software components, including source code, third-party code, open source libraries, dependencies, build tools, repositories, and deployment environments.[14] Each software product must be thoroughly analyzed for vulnerabilities before deploying it in an operational environment. Analysis activities need to address both custom-developed source code as well as third-party components, such as commercial and open source components and libraries, that are integrated into the product.

For example, the *fail-safe defaults* principle needs to be considered when applying a software update to an existing software application. A software update modifies an existing application to provide improvements, bug fixes, new features, or security patches. In an EIT environment, software patches are often automatically downloaded and installed to protect the enterprise from potential vulnerabilities that the update is addressing. However, applying a software patch without testing it could introduce unintended consequences to the enterprise, such as system instability, application conflicts, unexpected downtime, data loss, and disruption of business operations. As a result, software patches should not be deployed to operational systems by default. Patches need to be analyzed in a test or pre-production environment prior to being granted access to the operational environment.

The *fail-safe defaults* principle should be considered when implementing identity, credential, and access management (ICAM). In EIT environments, ICAM capabilities establish a centralized system to manage user identities, credentials, and access rights across an organization’s systems and networks. Typically, organizations implement a federated approach for ICAM, which is hosted in a cloud environment. As a result, there is no restriction on computing, memory, storage, or network capabilities. In an EIT environment, a full range of ICAM capabilities are available and can be used to manage access to an organization’s IT resources. The *fail-safe defaults* principle can be used to guide how an organization implements its ICAM capabilities and controls access to its resources.

For weapon system platforms, stakeholders must assess zero trust requirements and tradeoffs related to the principle of *fail-safe defaults*, particularly for provisioning new users, assigning role-based access privileges, and managing software updates. For example, implementing the concept of *no access by default* reduces the chances of sensitive data and resources being accessed by unauthorized users. However, if users unexpectedly need access to information and resources during mission execution (e.g., through dynamic reallocation of personnel), the application of the *fail-safe defaults* principle could prevent those users from accessing the information and resources they need to carry out their assignments. As a result, the mission could be put at risk.

[14] The software components included in a product should be documented in that product’s software bill of materials (SBOM). An SBOM typically contains a detailed list of all the software components, including libraries, frameworks, and dependencies, used within a software application along with information like their versions, licenses, origins, and any known vulnerabilities. The SBOM essentially provides a comprehensive inventory of the “ingredients” that make up the software, allowing for a better understanding of potential security risks and compliance issues within the software supply chain.

The limitations of tactical environments also need to be considered. For example, a military operation might occur in a denied, disrupted, intermittent, and limited (DDIL) environment,[15] where ICAM capabilities may be limited or non-existent. A weapon system in a DDIL environment would employ role-based access for users, where access privileges for users are preplanned. The application of the *fail-safe defaults* principle in DDIL environments is relevant when analyzing appropriate access privileges.

From a security perspective, the *fail-safe defaults* principle is a leading practice. However, its application in weapon system environments requires analysis and tailoring based on the mission and the associated opportunities and risks. Weapon system stakeholders should assess zero trust requirements and tradeoffs related to the principle of *fail-safe defaults* when making implementation decisions.

9.2 Questions

Can requirements change dynamically during mission execution (i.e., when the weapon system is supporting an operational mission), where application of fail-safe defaults could pose a risk to the mission?

- Are changes considered for both users and non-person entities (e.g., applications, systems, devices)?

Are there modes of weapon system operation where the application of fail-safe defaults could pose a risk to the mission?

- Are the risks of applying fail-safe defaults considered for both users and non-person entities (e.g., applications, systems, devices)?

Are there instances when users or non-person entities (e.g., applications, systems, devices) will unexpectedly need elevated privileges to support the weapon system's mission?

Are practices and procedures in place for assigning access rights to new users and devices?

- Are IT templates defined and used to provide access rights for new users of a weapon system?
- If the weapon system uses role-based access, are the different roles provided default rights or is a fail-safe defaults approach used for new users?
- When a new device is connected to a weapon system's network, can it access any resource within the weapon system?

Are fail-safe defaults considered when managing supply chain risks?

- If there is an update to a software product used in a weapon system, is it automatically updated?

[15] A DDIL environment is one in which internet access is unreliable, intermittent, or unavailable. These conditions limit real-time communication and data transfer, making it difficult to coordinate actions and understand the operational environment. As a result, DDIL environments can have an adverse impact on military operations, including intelligence gathering, decision making, and situational awareness.

- Has sufficient analysis been performed to establish access rights for software updates?
- Have software updates been tested to identify adverse consequences they might trigger (e.g., system instability, application conflicts, unexpected downtime, data loss, disruption of business operation)?

Have preplanned access privileges been established for environments where internet access is unreliable, intermittent, or unavailable (e.g., DDIL environments)?

Is access to the weapon system's resources and information denied unless permission is granted explicitly?

- Are decisions for implementing fail-safe defaults based on a review of risks, tradeoffs, and operational requirements?
- Has sufficient analysis been performed to establish default access rights for the weapon system's resources and information?

What workarounds or changes related to the *fail-safe defaults* principle might be needed in the weapon system's operational mission environment?

10 Complete Mediation

The security principle of *complete mediation* states that every access request to a resource must be checked every time, ensuring that unauthorized access is prevented. The following is a key definition of *complete mediation*:

- "Every access to every object must be checked for authority. This principle, when systematically applied, is the primary underpinning of the protection systems. It forces a system-wide view of access control, which in addition to normal operation includes initialization, recovery, shutdown, and maintenance. It implies that a foolproof method of identifying the source of every request must be devised. It also requires that proposals to gain performance by remembering the result of an authority check be examined skeptically. If a change in authority occurs, such remembered results must be systematically updated" [Saltzer 1975].

Complete mediation requires that each access to an IT resource is checked against the security policy to ensure that it is allowed. The access operation must be intercepted and determined to be acceptable before a resource can be accessed. Allowing automatic access to resources in a security zone or enclave once a user has gained access to that security zone or enclave violates the *complete mediation* principle. Access to resources must be checked every time.

Complete mediation is related to the *scrutinize explicitly* principle, which requires verifying and authenticating each access request from users, applications, systems, and devices based on available data.

10.1 Analysis

In an EIT environment, the following zero trust tenets are relevant to *complete mediation* [NIST 2020]:

- "Access to individual enterprise resources is granted on a per-session basis. Trust in the requester is evaluated before the access is granted."
- "All resource authentication and authorization are dynamic and strictly enforced before access is allowed. This is a constant cycle of obtaining access, scanning and assessing threats, adapting, and continually reevaluating trust in ongoing communication."

Identity, credential, and access management (ICAM) and asset management are services used in EIT environments to implement *complete mediation*. These services continually monitor user transactions and authenticate and authorize each transaction in accordance with the organization's security policy. Technologies that support ICAM and asset management services must be implemented correctly to comply with the *complete mediation* principle. Examples of these support technologies include caching and single sign-on (SSO).

- *Caching* is a technique that stores data in a temporary, high-speed storage area (i.e., the cache) to improve a system's performance. It enables faster retrieval of data that is frequently accessed rather than requiring repeated, slower access to the original data source. Designers must be careful when using caching to implement security mechanisms, such as authentication and access control. Implementing caching in accordance with the *complete mediation* principle requires checking every access to a cached object for authorization, even if the object is already in the cache. Using cached results of previously successful access grants for subsequent access requests violates the principle of *complete mediation*.

- *SSO* is an authentication method that enables users to securely authenticate with multiple applications, websites, and data repositories by using just one initial set of credentials. SSO itself is not inherently a violation of the *complete mediation* principle. However, an SSO implementation must be designed and managed carefully to ensure that every access to a resource is checked against the security policy. If SSO is not implemented correctly, it might allow a user to access multiple resources after a single authentication, bypassing individual access checks. In this way, the SSO implementation can violate the *complete mediation* principle.

Weapon system stakeholders must assess the tradeoffs associated with implementing the principle of *complete mediation* within the system. Stakeholders must evaluate the performance versus security requirements for weapon systems. Checking each transaction against the security policy before providing access consumes IT resources and can introduce latency, which can adversely affect the mission. The tradeoff analysis must consider the weapon system's role within the missions it supports, its internal processing requirements, and its interface requirements with other systems.

10.2 Questions

Has a security policy been established for mediating access to the weapon system's resources?

- How is the security policy for mediating access implemented in the weapon system?
- Can the security policy for mediating access be dynamically updated?

Does the weapon system monitor and control all access requests?

- Are security policies and access lists used to validate access to the weapon system's resources?
- Are all access requests to the weapon system's resources checked against the security policy before access is granted or denied?
- Does the weapon system have a logging capability that records the results of access requests?

Does the weapon system's architecture provide a mechanism for mediating all access to its resources?

- Does the weapon system have the capability to monitor and log access requests?
- Does the weapon system include enclaves where access rights cannot be checked?
- How does the weapon system track access to its data and resources if it does not include monitoring and logging capabilities?

What workarounds or changes related to the *complete mediation* principle might be needed in the weapon system's operational mission environment?

11 Open Design

The security principle of *open design* states that a system's security should not rely on the secrecy of its design or implementation. A system's security risks can be managed even if its architecture and algorithms are publicly known. The principle of *open design* states that systems should be designed in a manner that enables them to be easily inspected, analyzed, and modified by anyone with the necessary skills and knowledge. The following is a key definition of *open design* from a source document:

- "The design should not be secret. The mechanisms should not depend on the ignorance of potential attackers, but rather on the possession of specific, more easily protected, keys or passwords. This decoupling of protection mechanisms from protection keys permits the mechanisms to be examined by many reviewers without concern that the review may itself compromise the safeguards. . . . Finally, it is simply not realistic to attempt to maintain secrecy for any system which receives wide distribution" [Saltzer 1975].

Open design incorporates the following concepts:

- *transparency*—The architecture of a system should be accessible and understandable by security professionals. Security should not depend on keeping the system's architecture secret.
- *available documentation*—Design documents and specifications should clearly explain the system's architecture, algorithms, and security controls, making them available for review by security analysts.
- *external reviews*—External security experts should be tasked with reviewing the system's architecture to identify and address vulnerabilities and weaknesses in the system.

While openness is important for managing a system's security, sensitive information needs to be protected. For example, secret keys and implementation details that could be exploited by malicious actors should not be made public.

11.1 Analysis

The strength of a system's security should not depend on keeping the design hidden from potential attackers, a concept known as *security through obscurity*.[16] *Open design* is a core principle of many technologies implemented in EIT environments.

The principle of *open design* requires implementing well-established standards, leading practices, and transparent implementation details. It promotes the concept that security controls should be implemented in a way that enables them to be easily inspected, analyzed, and modified by anyone with the necessary skills and knowledge. In general, a system's architecture, algorithms, and

16 In the context of security engineering, *security though obscurity* is the idea that a system and its information can be protected when it is difficult to access or comprehend. This concept relies on making the design details and inner workings of a system less visible, reducing the likelihood of unauthorized access. Closed designs do not provide security over time because design secrets will likely be discovered and exploited by malicious actors, especially those who are motivated and highly skilled.

security controls should be documented and made available to security experts and analysts as appropriate. Making designs open and available enables developers and security experts to review a system's architecture and code, identify potential vulnerabilities, and recommend improvements.

Overall, the principle of *open design* can lead to more secure systems because it leverages the collective knowledge and expertise of the security community. For example, the open source community encourages its members to find and correct vulnerabilities and strengthen the overall levels of security in open source software (OSS) products. In this way, the open source community applies the principle of *open design* to share information about OSS products and improve their security.

System stakeholders in EIT environments routinely participate in security communities and forums to share knowledge, learn from others, and stay updated on emerging threats and best practices. This is an important practice for protecting an organization's IT resources. However, while information sharing is important when managing security in EIT environments, some sensitive information needs to be protected. For example, secret keys that could be exploited by malicious actors should not be made public.

The principle of *open design* is an important security principle in EIT environments. However, weapon system stakeholders need to assess the tradeoffs between releasing design information and restricting its disclosure. Many technologies in weapon systems provide a military advantage and promote survivability objectives. For example, critical program information (CPI) refers to information that could undermine U.S. military preeminence or technological advantage on the battlefield if compromised. Examples of CPI include technical specifications, design schematics, operational procedures, and supply chain data. As a result, opportunities to apply the principle of *open design* could be limited in weapon systems. Programs need to strike a balance between the principle of *open design* and the need to protect a weapon system's information.

11.2 Questions

To what extent can information about the weapon system's architecture and technology be shared openly?

- What information related to the weapon system has been classified as critical program information (CPI)? What are the protection requirements for the system's CPI?
- To what extent are well-established and widely reviewed security technologies (e.g., cryptographic algorithms and standards) implemented in the weapon system?
- Does the weapon system require unique security features (e.g., encryption technologies) whose specifications are not broadly available?
- Is there a need to restrict access to the weapon system's architecture, algorithms, and security controls?

Are the personnel conducting security reviews and assessments of the weapon system provided the information they need?

- What information about the weapon system is available to independent review teams?

- Is documentation for the weapon system available in peer reviews and system security assessments?
- Do classification issues restrict communication about the weapon system's architecture and technologies during reviews and assessments?

Is there a need to limit sharing technical information related to the weapon system?

- What information about the weapon system is restricted (i.e., Confidential, Secret, Top Secret)?
- What information about the weapon system can be discussed in open forums (e.g., security communities, discussion groups)?
- What practices and standards are used for sharing information about the weapon system's vulnerabilities and weaknesses?
- Are appropriate personnel aware of information-sharing policies and guidelines for the weapon system?

What changes or modifications to the information sharing and protection practices for the weapon system should be considered?

- What information might be more openly shared?
- What information should remain restricted?

What workarounds or changes related this the *open design* principle might be needed in the weapon system's operational mission environment?

12 Separation of Privilege

The principle of *separation of privilege* states that a system should not grant permission based on a single condition. Systems and programs granting access to resources should do so only when more than one condition is met. This practice provides fine-grained control over the resource as well as additional assurance that each access is authorized.

The following is a key definition of *separation of privilege* from a source document:

- "Where feasible, a protection mechanism that requires two keys to unlock it is more robust and flexible than one that allows access to the presenter of only a single key. The reason is that, once the mechanism is locked, the two keys can be physically separated and distinct programs, organizations, or individuals made responsible for them. From then on, no single accident, deception, or breach of trust is sufficient to compromise the protected information" [Saltzer 1975].

Implementing *separation of privilege* improves security by limiting access to data and resources, reducing the impact of potential breaches, and simplifying compliance efforts. Controlling access to data and resources also helps to reduce the attack surface, mitigate the impact of insider threats, and limit the lateral movement of attackers within an EIT environment.

Separation of privilege is often implemented alongside the principle of *least privilege*, which states that users should have access only to the resources they need to perform their job duties.

12.1 Analysis

In an EIT environment, *separation of privilege* ensures that no single entity possesses all the necessary permissions to compromise the security of a system or access critical resources. As a result, no single user will have complete control of all the elements of a critical function or system. Different roles and access levels are assigned to individuals, where one person might be responsible for initiating a transaction, another is responsible for approving it, and a third is responsible for recording it. Performing sensitive functions, such as acquiring root access, may require two or more individuals. This practice ensures that users fulfill their duties without exposing sensitive data or making unintended errors.

Non-person entities (e.g., applications, systems, devices) are granted access to resources only when more than one condition is met, which minimizes potential damage from breaches or misuse. For example, an attacker will need to defeat multiple safeguards to gain complete access to desired resources. In addition, software design practices recommend separating programs into parts. Each part requires certain privileges to perform a task. This design approach helps to ensure that an attack will be confined to one part of a software program; other parts cannot be exploited because access to those parts is restricted.

Separation of privilege can be enforced either statically (by defining roles that cannot be executed by one person) or dynamically (by enforcing it at access time). An example of dynamic enforcement is the *two-person rule*, where one authorized user initiates an operation, and a second authorized user is required to complete it. The two-person rule requires the approval of two authorized individuals for critical tasks [Sandhu 1994].

Traditional security practices apply *separation of privilege* at the system or network level, such as separating development environments from production environments. Zero trust requires a more granular approach for applying *separation of privilege* by requiring that all identities and resources be segmented from one another. This approach enables precise customization of access permissions, ensuring that users, applications, systems, and devices have access only to essential resources.

Systems can be used to automate the process of granting, revoking, and monitoring access in EIT environments to ensure that *separation of privilege* is enforced consistently across an organization. Automated systems can quickly adjust permissions in real time based on predefined policies, thereby reducing administrative overhead and improving efficiency. If it is set up appropriately, automation can enhance efficiency, accuracy, and security by streamlining user provisioning, role management, and compliance monitoring.

Weapon system stakeholders must assess zero trust requirements and tradeoffs related to *separation of privilege*. Weapon systems often operate in real time. Security checks and access control mechanisms in real-time systems need to be designed carefully to avoid disrupting operations and introducing latency. A thorough risk analysis will help stakeholders balance zero trust and mission requirements associated with *separation of privilege* by examining the associated risks and tradeoffs.

12.2 Questions

What are the appropriate access levels for different personnel or roles?

- How much power should administrative accounts have? How should administrative accounts work together?
- Are there any types of superuser vulnerabilities that could be eliminated by enforcing least privilege and separation of privilege?

Can separation of privilege be implemented in the weapon system?

- Which roles, responsibilities, and tasks need to be separated?
- Which of the weapon system's subsystems or components should be isolated from one another?
- Which functions of the weapon system require a granular approach for implementing separation of privilege (e.g., institute checks, prevent undesirable consequences)?
- Does the weapon system provide critical capabilities that require more than one permission (e.g., two-person rule)?

Are there cases where separation of privilege could potentially become constraining?

- Are controls being considered to prevent individuals from finding workarounds that circumvent perceived difficulties with access rules?

- In what cases would a two-person control process or transaction fail and require an override (e.g., one individual is out sick or on leave)?
- Is a contingency plan in place to account for when individuals are unavailable or their roles change?

Are practices and processes in place for monitoring and maintaining separation of privilege in the weapon system?

- Are user or service account transactions monitored and logged for auditability and future analysis?
- Are access permissions for the weapon system reviewed and updated periodically?
- Are the weapon system's subsystems and components monitored for suspicious activity?

Can automation be implemented in the weapon system to control access (e.g., granting, revoking, monitoring)?

- What is the threshold (e.g., number of users, varying levels of access) before manual access management becomes too complex, becomes prone to human error, and begins to lose effectiveness?
- Are there any cases where an automated access management system would impede user productivity or discourage users from adopting the system?
- Is there a dedicated team that can manage an automated access management system, providing the ongoing attention, review, and refinement necessary to ensure that it remains effective?

Are there cases where a failure of a separation of privilege policy could result in something catastrophic?

What workarounds or changes related to the *separation of privilege* principle might be needed in the weapon system's operational mission environment?

13 Minimize Secrets

The *minimize secrets* principle focuses on limiting the number and scope of secrets that are accessible to users and systems. Examples of secrets are digital credentials, passwords, application programming interface (API) keys,[17] encryption keys, secure shell (SSH) keys,[18] and tokens used for authentication and access control. Key definitions of *minimize secrets* from source documents include the following:

- "Minimize secrets: Because they probably won't remain secret for long. Following this principle has the following additional advantage. If the secret is compromised, it must be replaced; if the secret is minimal, then replacing the secret is easier" [Saltzer 2009].
- "This thoughtful addition to the list could be prone to misunderstanding. Secrets should be few and changeable, but they should also maximize entropy, and thus increase an attacker's work factor. The opposite situation clearly poses a problem, because each secret increases a system's administrative burden. For instance, a fighter jet project designed in the late 1990s required dozens of separately managed crypto keys to comply with data separation requirements that had been added piecemeal. The result was a management nightmare" [Smith 2012].

The *minimize secrets* principle requires that secrets (1) be few and easily interchangeable, (2) have a high degree of unpredictability, and (3) be minimal in complexity. When compromised, secrets can lead to attacks or breaches, which is why it is important to manage them properly. The *minimize secrets* principle reduces security risk by limiting the amount of sensitive information collected, stored, and accessed, which helps to minimize potential consequences from breaches and improve overall data security.

The security principle of *minimize secrets* is often applied in conjunction with the principles of *least privilege*, *open design*, and *complete mediation*. Applying these principles together significantly enhances an organization's security posture by reducing attack surfaces, limiting the impacts of security breaches, and promoting transparency and accountability, which ultimately produces a more secure EIT environment.

13.1 Analysis

The term *secrets* refers to any confidential data—like passwords, API keys, or encryption keys—that need protection from unauthorized access. Zero trust involves a higher level of sophistication

[17] An API key is a code used to identify an application or user and is used for authentication purposes. It acts as a credential, allowing an API provider to verify the identity of a requesting entity and control access to services. API keys are typically included in an API request to identify and authenticate the requesting user or application. Placing API keys in the request header is a common and preferred method for authentication. However, they can also be included in the API request body, depending on the API's design and security requirements. Regardless of where the key is placed, the connection must be secured with HTTPS to encrypt the data in transit.

[18] An SSH key comprises a pair of public and private keys for accessing and managing remote computers over an unsecured network via an encrypted communication channel (in accordance with the SSH protocol).

for applying the *minimize secrets* principle than simply employing basic password management practices. The broad range of secrets required in an EIT environment requires effective management of those secrets to prevent unauthorized access.

The *minimize secrets* principle suggests minimizing the number and complexity of secrets. This minimization is done by implementing effective management practices and controls. *Secrets management* is the practice of storing, managing, and controlling access to sensitive information like passwords and encryption keys. It is critical to manage keys and other secrets effectively because they provide access to data and resources across an EIT environment [DoD 2025].

Secrets management has become more critical in EIT environments as organizations shift to cloud and multi-cloud environments. In these environments, secrets are spread across multiple services and systems. The complexity of modern EIT environments makes it challenging to manage, control, and protect an organization's secrets. A centralized secrets management system can be used to track and manage an organization's secrets. Key topics related to secrets management include access control, secrets rotation, secure storage, auditing and monitoring, and automation.

- *Access control* is an essential part of a secrets management system, ensuring that only authorized users and applications can access passwords and keys. Role-based access control (RBAC), multi-factor authentication (MFA), and granular permissions[19] can be used to control access to secrets. Another consideration is granting access only when and for as long as a secret is needed. Secrets can be generated and used on demand rather than storing them permanently. They can also be based on short-lived credentials or tokens that expire after a specified time.
- *Secrets rotation* is the process of periodically updating or replacing secrets to reduce the potential impact of compromised secrets. Policies for updating or replacing secrets, including frequency, should be clearly defined and followed. EIT environments often employ automated processes for rotating secrets at regular intervals.
- *Secure storage*, such as secure vaults or encrypted files, is another aspect of a secrets management system that must be considered. The number of places where secrets are stored should be minimized. In general, secrets are sensitive data that need to be protected at rest and in transit using strong encryption algorithms.
- Comprehensive *auditing and monitoring* can effectively manage secrets when combined with auditing practices that track access, detect unauthorized activity, and ensure compliance with policy. Effectively monitoring and auditing secrets includes logging access attempts, reviewing logs for suspicious activity, and using tools to automate checks and notifications.
- *Automation* of secrets management is important from security, efficiency, and compliance perspectives. It reduces human error, streamlines workflows, and ensures the consistent application of security policies across all systems and environments. Automation is especially important in complex EIT environments.

[19] Granular permissions are access controls that are focused on specific resources and actions, helping to minimize the impact of a compromised account. This practice is associated with the principle of *least privilege*.

Weapon system stakeholders must assess zero trust requirements and tradeoffs related to the principle of *secrets management*. Weapon systems often have strict timing requirements. Implementing a secrets management system can introduce latency or processing complexity into accessing and managing secrets, which can potentially impact performance. Many weapon systems operate in dynamic and highly contested environments. These types of environments can make it difficult to manage secrets because they require flexible approaches. In addition, the real-time components of a weapon system often have complex dependencies between them. Identifying and minimizing the secrets needed by each component can be a challenge. Weapon system stakeholders should assess requirements and tradeoffs related to the *minimize secrets* principle when making implementation decisions.

13.2 Questions

Can a centralized secrets management system be implemented in the weapon system to track and manage secrets?

- What types of secrets need to be managed for the weapon system?
- Are policies for updating or replacing the weapon system's secrets (including frequency) defined and followed?
- Where will the secrets be stored (e.g., secure vaults, encrypted files)?
- Are the weapon system's secrets protected at rest and in transit using strong encryption algorithms?
- Will the latency introduced by secrets management practices adversely affect mission performance requirements?

Does the weapon system ensure that only authorized users and applications can access passwords and keys?

- Does the weapon system implement mechanisms to control access to secrets, such as role-based access control (RBAC), multi-factor authentication (MFA), and granular permissions?
- Are secrets generated and used on demand (i.e., dynamic secrets) rather than storing them permanently?
- Are secrets granted only when and for as long as they are needed (i.e., time limited)?

Does the weapon system monitor and audit the use of secrets to detect unauthorized activity and ensure compliance with policy?

- Are monitoring and auditing practices considered for logging access attempts and reviewing logs for suspicious activity?

Is automation implemented in the weapon system to facilitate secrets management?

- Are secrets management tasks (e.g., creation, rotation, revocation) automated to reduce human error and improve efficiency?
- Will the latency introduced by implementing automation and analytics for secrets management adversely affect mission performance requirements?

What workarounds or changes related to the *minimize secrets* principle might be needed in the weapon system's operational mission environment?

14 Summary and Conclusions

A zero trust architecture is a security model where every user, application, system, and device is untrusted by default. Each request to access computing resources must be authenticated dynamically before access is granted. Zero trust is based on the core concept that all networks are potentially compromised, so no entity should be trusted without verification. This philosophy runs counter to the traditional cybersecurity model for EIT environments, where measures and technologies are employed to protect an organization's systems and networks from unauthorized access by establishing a secure boundary between the internal and external networks.

14.1 Study Summary

Much of the published zero trust guidance focuses on applying zero trust concepts to EIT environments. However, the DoD is beginning to implement zero trust in weapon systems, which generally have different requirements than EIT systems. Guidance for tailoring and adapting zero trust concepts to weapon system platforms is needed by DoD programs responsible for implementing zero trust. The SEI study documented in this report takes an initial step toward providing this guidance.

Security and zero trust principles are important because they provide a foundation for developing, operating, and maintaining secure systems and protecting data. These principles define a basic framework for managing security risks in systems, applications, and processes. By ensuring that these principles are followed, system stakeholders can have a reasonable degree of assurance that security risks are being managed and that their systems are positioned to achieve mission success.

Security and zero trust principles were originally designed for general-purpose computing systems, such as those found in EIT environments. As part of this study, the SEI team analyzed how to tailor EIT-focused security and zero trust principles to weapon system platforms. The analysis includes understanding how each principle is applied in EIT environments and establishing factors to consider when tailoring the principle to weapon system environments. A key aspect of the analysis was developing a set of questions for each principle that can be used to evaluate tradeoffs and tailoring options when applying that principle to weapon system environments.[20]

14.2 High-Level Conclusions

In addition to the detailed analysis of selected security and zero trust principles, the SEI team identified several high-level conclusions based on the information collected. The team analyzed data across the principles, looking for patterns and insights. As a result, the team identified the following high-level conclusions:

- **The first step when tailoring principles to a specific weapon system is to understand the environment where the weapon system is deployed.** The significance of each principle is

[20] Find tailoring questions for the principles in Sections 6-13 of this report.

best understood when it is viewed from within the environment where it is applied. Four aspects of a weapon system's environment are particularly important to understand: mission context, system attributes, threat environment, and tradeoff space. By understanding a weapon system's environment, stakeholders can tailor security principles to a weapon system's unique circumstances, thereby enhancing the system's ability to achieve its mission.

- **Weapon system stakeholders should tailor solutions and make design choices for addressing security principles based on a systematic analysis of the tradeoffs among the system's requirements and objectives.** A tradeoff space refers to the range of possible solutions or design choices where different requirements of a system are in conflict. Stakeholders must analyze how competing requirements relate to one another to determine where opportunities in one area might produce risks or problems in another. The goal is to balance security requirements with performance and interoperability requirements as well as program cost and schedule objectives. Stakeholders can then tailor solutions and make design choices that address security principles based on a systematic analysis of the tradeoffs.
- **The security and zero trust principles provide a framework for making security decisions and implementing security controls to enable mission assurance through effective risk management.** Mission assurance focuses on ensuring the continued function and resilience of a weapon system's critical assets and capabilities during mission execution. The security and zero trust principles provide a framework for making security decisions and implementing security controls that will reduce the vulnerabilities and security risks in weapon systems. An effective risk management practice helps stakeholders proactively implement strategies to protect a weapon system's mission-critical functions and ensure continuity of operations even in highly contested mission environments.
- **Significant relationships and dependencies among the security and zero trust principles provide an opportunity to leverage security resources, practices, and controls to increase the potential for mission success.** Mutually exclusive concepts have no overlap or common ground; they are distinct and separate from one another. The nine security and zero trust principles analyzed in this study are not mutually exclusive. There are significant relationships and dependencies among them. These relationships and dependencies provide opportunities to leverage security resources, practices, and controls to enhance a weapon system's security posture, manage security risks more effectively, and increase the potential for mission success.

14.3 Final Thoughts

Zero trust is another phase in the ongoing evolution of security strategies needed to manage emerging threats and deploy new technologies across the systems lifecycle. Mission environments are dynamic and require ongoing tuning, refinements, and improvements to ensure that resources and risks are managed effectively. Therefore, effective management in these environments requires monitoring risks and strategies closely and being prepared to adapt when necessary.

Zero trust assumes that no user or device is inherently trustworthy, so each access attempt requires verification and authorization. This shift in philosophy represents a move away from traditional perimeter-based security models and introduces a significant change in implementing authentication, authorization, and security controls. Fortunately, the principles covered in this report provide a strong foundation for managing the transition to zero trust in weapon system environments.

Principles are basic ideas or concepts that explain how something is supposed to work. They provide a bridge between theory and practice and help to make abstract ideas actionable. While principles are based on theories, they are more concrete and specific than theories and provide a framework for their implementation. The SEI's study of security and zero trust principles provides foundational content that can help inform the development of a zero trust implementation roadmap for weapon systems. Building an implementation roadmap is an essential next step for applying the ideas and concepts embodied in a set of principles, such as those included in this study.

Appendix: Zero Trust Tailoring Questions by Topic

The SEI team developed a set of questions that can be used to evaluate tradeoffs and tailoring options when applying each principle to weapon system environments. This appendix categorizes the tailoring questions by topic, providing an alternative means of reviewing the content. Topic categories were selected by identifying common themes among the questions.

Each topic below includes the main tailoring questions and associated sub-questions that were identified during the SEI analysis. A tag links each main tailoring question to the principle from which it was derived. For additional context about any question in this appendix, review the description and analysis information for the related principle as documented in Sections 6-13 of this report.

Access Control

Can access permissions to the weapon system's capabilities change in real time? [Least Privilege]

- Are access permissions considered for both users and non-person entities (e.g., applications, systems, devices)?

Are there instances when users or non-person entities (e.g., applications, systems, devices) will unexpectedly need elevated privileges to support the weapon system's mission? [Least Privilege and Fail-Safe Defaults]

Are there instances where access privileges need to persist beyond a session (i.e., not limited by time and scope)? [Least Privilege]

Which applications, systems, and devices can access the weapon system during mission execution? [Least Privilege]

- How are these dependencies managed?

Which applications, systems, and devices can access the weapon system during system maintenance activities? [Least Privilege]

- How are these dependencies managed?

Are practices and procedures in place for assigning access rights to new users and devices? [Fail-Safe Defaults]

- Are IT templates defined and used to provide access rights for new users of a weapon system?
- If the weapon system uses role-based access, are the different roles provided default rights or is a fail-safe defaults approach used for new users?
- When a new device is connected to a weapon system's network, can it access any resource within the weapon system?

Have preplanned access privileges been established for environments where internet access is unreliable, intermittent, or unavailable (e.g., DDIL environments)? [Fail-Safe Defaults]

Is access to the weapon system's resources and information denied unless permission is granted explicitly? [Fail-Safe Defaults]

- Are decisions for implementing fail-safe defaults based on a review of risks, tradeoffs, and operational requirements?
- Has sufficient analysis been performed to establish default access rights for the weapon system's resources and information?

What are the appropriate access levels for different personnel or roles? [Separation of Privilege]

- How much power should administrative accounts have? How should administrative accounts work together?
- Are there any types of superuser vulnerabilities that could be eliminated by enforcing least privilege and separation of privilege?

Does the weapon system ensure that only authorized users and applications can access passwords and keys? [Minimize Secrets]

- Does the weapon system implement mechanisms to control access to secrets, such as role-based access control (RBAC), multi-factor authentication (MFA), and granular permissions?
- Are secrets generated and used on demand (i.e., dynamic secrets) rather than storing them permanently?
- Are secrets granted only when and for as long as they are needed (i.e., time limited)?

Authentication and Authorization

Can every access request, regardless of location or source, be authenticated and authorized using multiple attributes (e.g., multi-factor authentication, device health checks, user context)? [Presume Breach and Scrutinize Explicitly]

- Will latency introduced by authentication capabilities adversely affect mission performance requirements?

Architecture

Is dividing the system's architecture into small, isolated segments (to restrict lateral movement of attackers within the network) practical for the weapon system? [Presume Breach]

- Will a highly segmented architecture have an impact on operational performance and mission success?
- Is a highly segmented architecture practical for weapon systems that are built on commercial or legacy platforms?

Does the weapon system's architecture provide a mechanism for mediating all access to its resources? [Complete Mediation]

- Does the weapon system have the capability to monitor and log access requests?

- Does the weapon system include enclaves where access rights cannot be checked?
- How does the weapon system track access to its data and resources if it does not include monitoring and logging capabilities?

Automation and Analytics

Are automation and analytics implemented to manage large volumes of data for collecting, analyzing, and correlating security data for the weapon system and its subsystems/components? [Presume Breach and Scrutinize Explicitly]

- Will the processing needed to collect, analyze, and manage security data adversely affect mission performance requirements (e.g., by introducing latency)?
- Can dynamic policy adjustments for the weapon system and its subsystems/components introduce interoperability or performance risks to the mission?

Can automation be implemented in the weapon system to control access (e.g., granting, revoking, monitoring)? [Separation of Privilege]

- What is the threshold (e.g., number of users, varying levels of access) before manual access management becomes too complex, becomes prone to human error, and begins to lose effectiveness?
- Are there any cases where an automated access management system would impede user productivity or discourage users from adopting the system?
- Is there a dedicated team that can manage an automated access management system, providing the ongoing attention, review, and refinement necessary to ensure that it remains effective?

Is automation implemented in the weapon system to facilitate secrets management? [Minimize Secrets]

- Are secrets management tasks (e.g., creation, rotation, revocation) automated to reduce human error and improve efficiency?
- Will the latency introduced by implementing automation and analytics for secrets management adversely affect mission performance requirements?

Device Security Check

Is it feasible to perform a device posture check to verify a device's current security status before granting access to a weapon system's resources? [Presume Breach and Scrutinize Explicitly]

- Can a device's security posture be assessed continually to ensure it meets policies for accessing sensitive data?
- Can devices be checked during military operations to ensure that they are up to date with patches and have appropriate security software?

Encryption

Should encryption be implemented in the weapon system to protect data at rest and in transit? [Presume Breach]

- Will latency introduced by encryption capabilities adversely affect mission performance requirements?

Fail-Safe Defaults

Are there conditions where a deny-access state may not be a safe default state? [Presume Breach]

- Are there circumstances where an operator absolutely must be able to take control of a weapon system?
- Does design guidance describe when deny by default could be an unsafe condition?

Can requirements change dynamically during mission execution (i.e., when the weapon system is supporting an operational mission), where application of fail-safe defaults could pose a risk to the mission? [Fail-Safe Defaults]

- Are changes considered for both users and non-person entities (e.g., applications, systems, devices)?

Are there modes of weapon system operation where the application of fail-safe defaults could pose a risk to the mission? [Fail-Safe Defaults]

- Are the risks of applying fail-safe defaults considered for both users and non-person entities (e.g., applications, systems, devices)?

Are fail-safe defaults considered when managing supply chain risks? [Fail-Safe Defaults]

- If there is an update to a software product used in a weapon system, is it automatically updated?
- Has sufficient analysis been performed to establish access rights for software updates?
- Have software updates been tested to identify adverse consequences they might trigger (e.g., system instability, application conflicts, unexpected downtime, data loss, disruption of business operation)?

Information Sharing

To what extent can information about the weapon system's architecture and technology be shared openly? [Open Design]

- What information related to the weapon system has been classified as critical program information (CPI)? What are the protection requirements for the system's CPI?
- To what extent are well-established and widely reviewed security technologies (e.g., cryptographic algorithms and standards) implemented in the weapon system?
- Does the weapon system require unique security features (e.g., encryption technologies) whose specifications are not broadly available?

- Is there a need to restrict access to the weapon system's architecture, algorithms, and security controls?

Are the personnel conducting security reviews and assessments of the weapon system provided the information they need? [Open Design]

- What information about the weapon system is available to independent review teams?
- Is documentation for the weapon system available in peer reviews and system security assessments?
- Do classification issues restrict communication about the weapon system's architecture and technologies during reviews and assessments?

Is there a need to limit sharing technical information related to the weapon system? [Open Design]

- What information about the weapon system is restricted (i.e., Confidential, Secret, Top Secret)?
- What information about the weapon system can be discussed in open forums (e.g., security communities, discussion groups)?
- What practices and standards are used for sharing information about the weapon system's vulnerabilities and weaknesses?
- Are appropriate personnel aware of information-sharing policies and guidelines for the weapon system?

What changes or modifications to the information sharing and protection practices for the weapon system should be considered? [Open Design]

- What information might be more openly shared?
- What information should remain restricted?

Monitoring

Can continuous monitoring be implemented in the weapon system? [Scrutinize Explicitly]

- Can user activity be monitored in real time for suspicious behavior?
- Is it feasible to implement real-time security analytics to detect anomalies and potential threats during mission execution?

Does the weapon system monitor and control all access requests? [Complete Mediation]

- Are security policies and access lists used to validate access to the weapon system's resources?
- Are all access requests to the weapon system's resources checked against the security policy before access is granted or denied?
- Does the weapon system have a logging capability that records the results of access requests?

Are practices and processes in place for monitoring and maintaining separation of privilege in the weapon system? [Separation of Privilege]

- Are user or service account transactions monitored and logged for auditability and future analysis?
- Are access permissions for the weapon system reviewed and updated periodically?
- Are the weapon system's subsystems and components monitored for suspicious activity?

Does the weapon system monitor and audit the use of secrets to detect unauthorized activity and ensure compliance with policy? [Minimize Secrets]

- Are monitoring and auditing practices considered for logging access attempts and reviewing logs for suspicious activity?

Policy

Is a zero trust policy developed for the weapon system and its subsystems/components as appropriate? [Scrutinize Explicitly]

- Does the zero trust policy for the weapon system and its subsystems/components align with mission requirements?
- Could interoperability issues arise if the zero trust policy for the weapon system is not aligned with mission requirements?

Has a security policy been established for mediating access to the weapon system's resources? [Complete Mediation]

- How is the security policy for mediating access implemented in the weapon system?
- Can the security policy for mediating access be dynamically updated?

Are there cases where a failure of a separation of privilege policy could result in something catastrophic? [Separation of Privilege]

Response and Recovery

Have response and recovery plans been developed for the weapon system? [Presume Breach]

- Has an incident response plan for managing security events and incidents been developed and tested for the weapon system?
- Has a recovery plan for restoring a system and its data after a disruption been developed and tested for the weapon system?

Role Definition

Can roles and requirements for users or non-person entities (e.g., applications, systems, devices) change dynamically during mission execution (i.e., when the weapon system is supporting an operational mission)? [Least Privilege]

Secrets Management System

Can a centralized secrets management system be implemented in the weapon system to track and manage secrets? [Minimize Secrets]

- What types of secrets need to be managed for the weapon system?
- Are policies for updating or replacing the weapon system's secrets (including frequency) defined and followed?
- Where will the secrets be stored (e.g., secure vaults, encrypted files)?
- Are the weapon system's secrets protected at rest and in transit using strong encryption algorithms?
- Will the latency introduced by secrets management practices adversely affect mission performance requirements?

Separation of Privilege

Can separation of privilege be implemented in the weapon system? [Separation of Privilege]

- Which roles, responsibilities, and tasks need to be separated?
- Which of the weapon system's subsystems or components should be isolated from one another?
- Which functions of the weapon system require a granular approach for implementing separation of privilege (e.g., institute checks, prevent undesirable consequences)?
- Does the weapon system provide critical capabilities that require more than one permission (e.g., two-person rule)?

Are there cases where separation of privilege could potentially become constraining? [Separation of Privilege]

- Are controls being considered to prevent individuals from finding workarounds that circumvent perceived difficulties with access rules?
- In what cases would a two-person control process or transaction fail and require an override (e.g., one individual is out sick or on leave)?
- Is a contingency plan in place to account for when individuals are unavailable or their roles change?

Situational Awareness

Can the weapon system be monitored for security threats during mission execution? [Presume Breach]

- Is data collected, analyzed, and communicated to provide adequate situational awareness of the weapon system's threat environment?
- Is it feasible to implement real-time security analytics to detect anomalies and potential threats during mission execution?

User and Asset Inventory

Is it feasible to establish a comprehensive list of users, applications, systems, and devices for the mission that the weapon system supports? [Scrutinize Explicitly]

- Is the mission environment too dynamic to create a comprehensive user and asset inventory?
- Could interoperability issues arise if users, applications, systems, and devices not in the inventory are assigned to participate in the mission?

Workarounds

What workarounds or changes related to each principle might be needed in the weapon system's operational mission environment? [All Principles]

References

URLs are valid as of the publication date of this report.

[Alberts 2014]
Alberts, Christopher; Woody, Carol; & Dorofee, Audrey. *Introduction to the Security Engineering Risk Analysis (SERA) Framework.* CMU/SEI-2014-TN-025. DOI: 10.1184/R1/6574856.v1. Software Engineering Institute, Carnegie Mellon University. December 4, 2014. https://insights.sei.cmu.edu/library/introduction-to-the-security-engineering-risk-analysis-sera-framework/

[Alberts 2024]
Alberts, Christopher; Wallen, Charles M.; Woody, Carol; Bandor, Michael S.; & Merendino, Tom. *Security Engineering Framework (SEF): Managing Security and Resilience Risks Across the Systems Lifecycle.* CMU/SEI-2024-SR-022. DOI: 10.1184/R1/25029359. Software Engineering Institute, Carnegie Mellon University. December 12, 2024. https://insights.sei.cmu.edu/library/security-engineering-framework-sef-24sr022/

[DAU 2024]
Defense Acquisition University (DAU). *DAU Glossary of Defense Acquisition Acronyms and Terms*. DAU Website. March 28, 2024 [accessed]. https://www.dau.edu/glossary

[DISA 2022]
Defense Information Systems Agency (DISA) and National Security Agency (NSA) Zero Trust Engineering Team. *DoD Zero Trust Reference Architecture Version 2.0*. DISA. July 2022. https://dodcio.defense.gov/Portals/0/Documents/Library/(U)ZT_RA_v2.0(U)_Sep22.pdf

[DoD 2022a]
Department of Defense (DoD) Chief Information Officer (CIO) Zero Trust Portfolio Management Office (PfMO). *DoD Zero Trust Strategy.* DoD. October 21, 2022. https://dodcio.defense.gov/Portals/0/Documents/Library/DoD-ZTStrategy.pdf

[DoD 2022b]
Department of Defense (DoD). *Systems Engineering Guidebook.* DoD. February 2022. https://ac.cto.mil/wp-content/uploads/2022/02/Systems-Eng-Guidebook_Feb2022-Cleared-slp.pdf

[DoD 2022c]
Department of Defense (DoD). *Cyber Survivability Endorsement (CSE) Implementation Guide, Version 2.0*. DoD. August 22, 2022. https://www.dau.edu/cop/rqmt/documents/guide-cyber-survivability-endorsement-implementation-guide

[DoD 2023a]
Department of Defense (DoD) Office of the Under Secretary of Defense for Research and Engineering. *Department of Defense Mission Engineering Guide, Version 2.0.* DoD. October 1, 2023. https://ac.cto.mil/wp-content/uploads/2023/11/MEG_2_Oct2023.pdf

[DoD 2023b]
Department of Defense (DoD) Office of the Assistant of Secretary of Defense for Acquisition / Office of the Deputy Assistant Secretary of Defense for Platform and Weapon Portfolio Management. *Zero Trust: Analysis of the Applicability to Weapon Systems and Defense Critical Infrastructure, Version 1.0*. DoD. June 2023.

[DoD 2024]
Department of Defense (DoD) Chief Information Officer (CIO). *Department of Defense Zero Trust Overlays, Version 1.1*. DoD. June 2024. https://dodcio.defense.gov/Portals/0/Documents/Library/ZeroTrustOverlays.pdf

[DoD 2025]
Department of Defense (DoD). *Cloud Security Playbook: Volume 1.* DoD. February 11, 2025. https://dodcio.defense.gov/Portals/0/Documents/Library/CloudSecurityPlaybookVol1.pdf

[Dorofee 2003]
Dorofee, Audrey; Woody, Carol; Alberts, Christopher; Creel, Rita; & Ellison, Robert J. *A Systemic Approach for Assessing Software Supply-Chain Risk*. February 2003. https://insights.sei.cmu.edu/documents/439/2013_019_001_297385.pdf

[Hilburn 2023]
Hilburn, Tom; Fairley, Dick; & Squires, Alice. Software Engineering in the Systems Engineering Life Cycle. *SEBoK Wiki.* December 14, 2023 [accessed]. https://sebokwiki.org/wiki/Software_Engineering_in_the_Systems_Engineering_Life_Cycle

[NIST 2018]
National Institute of Standards and Technology (NIST). *Framework for Improving Critical Infrastructure Cybersecurity, Version 1.1.* NIST. April 16, 2018. https://doi.org/10.6028/NIST.CSWP.04162018

[NIST 2020]
National Institute of Standards and Technology (NIST). *Zero Trust Architecture*. NIST SP 800-207. NIST. August 2020. https://csrc.nist.gov/pubs/sp/800/207/final

[NIST 2021]
National Institute of Standards and Technology (NIST). *Developing Cyber-Resilient Systems: A Systems Security Engineering Approach.* NIST SP 800-160 Vol. 2 Rev.1. NIST. December 2021. https://csrc.nist.gov/publications/detail/sp/800-160/vol-2-rev-1/final

[NIST/DOC 2012]
Joint Task Force Transformation Initiative: National Institute of Standards and Technology (NIST) and U.S. Department of Commerce (DOC). *Guide for Conducting Risk Assessments.* NIST SP 800-30 Rev. 1. NIST. September 2012. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf

[NIST/DOC 2020]
Joint Task Force Interagency Working Group: National Institute of Standards and Technology (NIST) and U.S. Department of Commerce (DOC). *Security and Privacy Controls for Information Systems and Organizations.* NIST SP 800-53 R5. NIST. September 2020. https://nvl-pubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r5.pdf

[NSA 2021]
National Security Agency (NSA). *Embracing a Zero Trust Security Model*. NSA. February 2021. https://media.defense.gov/2021/Feb/25/2002588479/-1/-1/0/CSI_EMBRACING_ZT_SECURITY_MODEL_UOO115131-21.pdf

[Saltzer 1975]
Saltzer, J. H. & Schroeder, M. D. The Protection of Information in Computer Systems. *Proceedings of the IEEE.* Volume: 63. Issue 9. September 1975. Pages 1278–1308. https://ieeex-plore.ieee.org/document/1451869

[Saltzer 2009]
Saltzer, J. H. & Kaashoek, M. F. *Principles of Computer System Design.* Morgan Kaufmann. 2009. ISBN: 978-0-12-374957-4. https://doi.org/10.1016/C2009-0-20124-3

[Sandhu 1994]
Sandhu, R. S. & Samarati, P. Access Control: Principle and Practice. *IEEE Communications Magazine*. Volume 32. Issue 9. September 1994. Pages 40–48. https://doi.org/10.1109/35.312842

[Smith 2012]
Smith, Richard E. A Contemporary Look at Saltzer and Schroeder's 1975 Design Principles. *Security & Privacy.* Volume 10. Issue 6. November-December, 2012. Pages 20–25. https://ieeex-plore.ieee.org/document/6226346

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.			
1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE September 2025		3. REPORT TYPE AND DATES COVERED Final
4. TITLE AND SUBTITLE Tailoring Security and Zero Trust Principles to Weapon System Environments			5. FUNDING NUMBERS FA870225DB003
6. AUTHOR(S) Christopher Alberts, Rhonda Brown, Timothy Morrow, and Charles M. Wallen			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213			8. PERFORMING ORGANIZATION REPORT NUMBER CMU/SEI-2025-SR-013
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) SEI Administrative Agent AFLCMC/AZS 5 Eglin Street Hanscom AFB, MA 01731-2100			10. SPONSORING/MONITORING AGENCY REPORT NUMBER n/a
11. SUPPLEMENTARY NOTES			
12A DISTRIBUTION/AVAILABILITY STATEMENT Unclassified/Unlimited, DTIC, NTIS			12B DISTRIBUTION CODE
13. ABSTRACT (MAXIMUM 200 WORDS) Zero trust is a security model where every user, application, system, and device is untrusted by default, requiring verification and authorization for every access attempt. A key aspect of zero trust is the concept that today's infrastructures no longer have clearly defined perimeters. The movement to a zero trust philosophy changes how an organization implements its security strategy, driven by the need to manage evolving threats and technologies. Much of the available zero trust guidance focuses on applying zero trust concepts to enterprise information technology (EIT) environments. The Department of Defense (DoD) is on the path to implementing zero trust in weapon systems, which generally have different requirements than EIT systems. DoD stakeholders need guidance on how to tailor and adapt zero trust concepts to weapon system platforms. To address this need, the Software Engineering Institute (SEI) conducted a study that analyzed the applicability of foundational security and zero trust principles to weapon system environments. These principles define a framework for making security decisions and implementing security controls, enabling mission assurance through effective risk management. This report provides analysis results for nine security and zero trust principles included in the study.			
14. SUBJECT TERMS security, zero trust, weapons systems, real-time systems			15. NUMBER OF PAGES 61
16. PRICE CODE			
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. Z39-18 298-102