TRUSTWORTHY SECURE DESIGN

Engineering Security from the Inside Out

Dr. Ron Ross

CEO, RONROSSECURE, LLC
ISTS Fellow, Dartmouth College

Abstract

Cybersecurity is entering a period in which frontier artificial intelligence (AI) can increase the speed, scale, accessibility, and persistence of cyber operations while complex supply chains, cloud services, autonomous agents, cyber-physical systems, and accumulated technical debt expand the consequences of compromise. NIST SP 800-160, Volume 1, offers a timely engineering response: treat security as an emergent property of a system; begin with intended behaviors, system states, and conditions that can lead to loss; and apply the Security Design Order of Precedence (SecDOP) to eliminate or reduce loss potential through design before relying on engineered features, monitoring, procedures, or training. This paper discusses the NIST SP 800-160 concepts for trustworthy secure design as an integrated strategy, relates it to the security design principles and the system life cycle processes through which those concepts are executed, and connects trustworthy secure design to the concept of assurance. Its central argument is simple: organizations cannot operationally compensate forever for architectures that were never designed to control asset loss. The most consequential security decisions are often made before a security control is selected.

1. Introduction

The cyber threat environment is changing in ways that amplify an old structural weakness. Organizations have spent decades improving vulnerability management, identity protection, incident response, endpoint defense, system monitoring, and other essential operational capabilities. Yet many of the systems those capabilities protect were not designed from the outset to remain trustworthy secure when components fail, credentials are stolen, software is compromised, supply chains are subverted, or adversaries obtain an unexpected foothold.

Frontier AI raises the stakes because it can compress attacker timelines and lower the cost of searching large attack surfaces. Recent public analyses have warned that frontier models can make cyber operations faster, more scalable, and more accessible, while also strengthening defensive capabilities [1][2][3]. The engineering implication is not that every future attack will be autonomous or unprecedented. It is that defenders may have less time to compensate operationally for weaknesses embedded in system structure.

These developments amplify a longstanding systems security engineering concern. The compromise of a component should not automatically become a *system* compromise or *mission* loss. Protection strategies must be multidimensional, combining architectural, technical, operational, and resilience measures rather than depending on any single class of security controls. This becomes especially important for high-impact or high-consequence cyber-physical systems and critical AI infrastructure, where the consequences of compromise can extend well beyond individual components. As AI increases the speed and scale of cyber operations, closing the gap between what we know how to engineer and what organizations have actually built becomes increasingly urgent.

NIST SP 800-160 Volume 1 [4] is particularly relevant to this moment. The publication's purpose is broader than cybersecurity operations: it treats security as an *emergent* system property created through systems engineering. SP 800-160 concentrates that philosophy into a *design approach*: define what the system is supposed to do; identify the states and conditions that can lead to unacceptable loss; design the system to

prevent and limit those losses; anticipate that protective functions themselves can fail; and provide the structural, functional, and evidentiary basis for justified confidence in the resulting system [4].

This is not a rejection of security controls. SP 800-160 explicitly recognizes engineered security features and devices. The distinction is one of precedence and context. Security controls are most effective when they are integrated into a sound architecture rather than asked to compensate for a fundamentally weak one.

Frontier AI changes the speed and economics of cyber conflict. It does not repeal the laws of good engineering.

---

2. Integrated Design Strategy

SP 800-160 describes trustworthy secure design as a balanced and integrated approach for optimally protecting against asset loss. It is organized around four mutually reinforcing subjects: (1) the design approach for trustworthy systems; (2) design considering emergence; (3) the Security Design Order of Precedence; and (4) functional design considerations. SP 800-160 also makes two important connections explicit. First, its concepts and the security design principles together provide a basis for reasoning about a system and demonstrating *trustworthiness*. Second, trustworthiness and *assurance* needs inform the design of system components and the system.

The phrase *trustworthy secure design* is important. The objective is not merely to add security functionality. It is to give stakeholders confidence that conflicting capability needs, concerns, priorities, and constraints have been satisfied while the system continues to deliver acceptable performance. Security, therefore, participates in the same engineering trade space as mission capability, safety, reliability, availability, cost, schedule, size, weight, power, usability, and other system objectives.

SP 800-160 is also explicitly system life cycle oriented. Design principles, standards, specifications, patterns, policy models, cryptographic algorithms, protocols, strength-of-mechanism decisions, and knowledge of adversity are to be planned, scoped, and revisited throughout the engineering effort. That point matters because a trustworthy secure design can be weakened after initial development by integration choices, maintenance pathways, software updates, enabling systems, supply-chain substitutions, operational workarounds, or changes in mission use.

The most consequential security decisions are often made before a security control is selected.

---

3. Start with Behaviors, States, Conditions, and Loss

The SP 800-160 design approach seeks to establish and maintain the ability to deliver system capabilities at an acceptable level of performance while minimizing the occurrence and extent of asset loss. The system must provide intended behaviors and outcomes, avoid unintended behaviors and outcomes, prevent loss, and limit loss when prevention is not possible. This framing immediately shifts the security conversation from security controls to *system behavior*.

SP 800-160 identifies five iterative elements:

1. Define the intended behaviors and outcomes for the system.
2. Identify the system states and conditions that reflect those intended behaviors and outcomes.
3. Identify the system states and conditions that potentially lead to loss.
4. Select and alter the system design to prevent loss to the extent practicable and limit the loss that does occur.
5. Iterate the analysis to address how the functions intended to prevent or limit loss may themselves fail for intentional or unintentional reasons. [4]

This sequence is deceptively powerful. It asks engineers to describe security in specific terms that can be represented in requirements, architecture, state models, interfaces, functional allocations, constraints, and verification evidence. An unauthorized data transfer, for example, is not merely a system alert or a policy violation. It is a *system transition* into a condition that can produce asset loss. A privilege escalation is not simply a vulnerability class. It is a change in *system authority* that can enable behaviors the stakeholders did not authorize.

The approach also requires *margin* and *situational awareness*. Margin accounts for uncertainty and adversity; situational awareness supports accountability and helps detect pending or actual failure. SP 800-160 explicitly connects this capability to the *Anomaly Detection* and *Loss Margins* security design principles. The implication is that a trustworthy design must not only function correctly under nominal conditions. It must also provide enough information and operating margin to recognize when assumptions are being violated or the system is approaching a dangerous or unstable state.

SP 800-160 describes the steps in the design approach in the context of a *systems security engineering framework*: intended behavior, system states reflecting intended behavior, and states that can lead to loss define the *problem space*; engineering to prevent and limit loss is the *solution space* response [4]. That separation is useful because it discourages premature commitment to a favorite technology before the loss problem has been adequately defined. Figure 1 illustrates the problem and solution spaces in the systems security engineering framework.

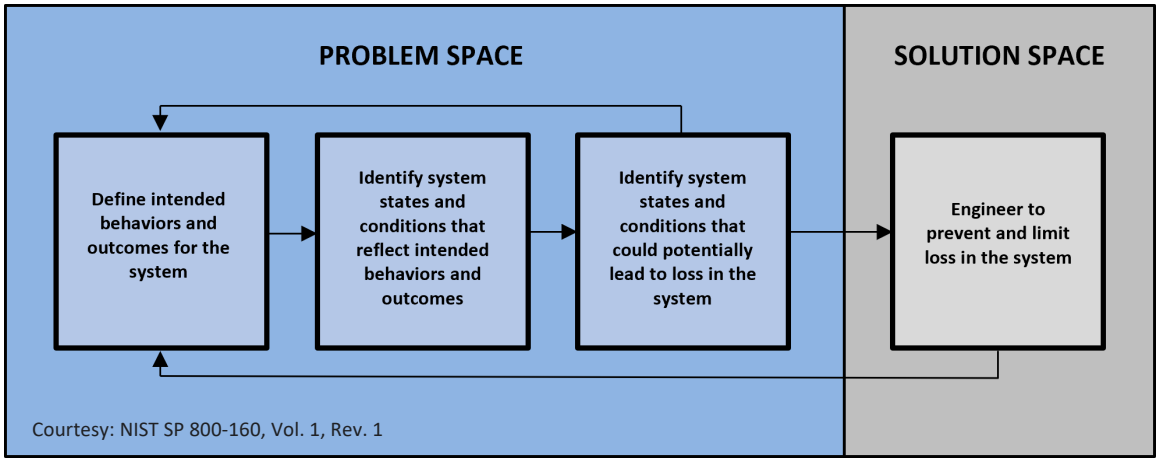

Figure 1. Design Approach in a Systems Security Engineering Framework [4]

4. A Balanced Strategy: Preemptive and Reactive Protection

SP 800-160 rejects a false choice between *prevention* and *response*. Trustworthy secure design requires both. The *preemptive* aspect anticipates conditions under which asset loss may occur and introduces

system features and actions before the loss. The *reactive* aspect recognizes that uncertainty is irreducible: unanticipated events, failures, and adverse consequences will occur despite planning. The reactive aspect is therefore not an admission that engineering failed. It is a proactive engineering decision to provide a capability for informed action after adverse conditions arise. The system is designed in advance to support containment, degraded operation, recovery, operator intervention, or other responses appropriate to the significance and context of loss.

This is particularly important for sophisticated adversaries and AI-enabled attacks. A protection strategy that assumes perfect prevention becomes increasingly fragile as attack speed, discovery capability, and persistence increase. A *balanced strategy* asks a different question: if prevention fails, what system behaviors remain possible, what authority remains available to the adversary, what boundaries still hold, what loss can occur, how quickly will the condition be detected, and what recovery path remains viable and trustworthy?

SP 800-160 supplies security design principles that help realize this balance. *Domain Separation*, *Least Sharing*, *Least Privilege*, *Mediated Access*, *Distributed Privilege*, *Minimal Trusted Elements*, *Defense in Depth*, *Protective Defaults*, and *Continuous Protection* strengthen the preemptive side. *Commensurate Response*, *Anomaly Detection*, *Protective Failure*, *Protective Recovery*, *Redundancy*, and *Loss Margins* strengthen the ability to recognize, contain, respond to, and recover from adversity. The principles are not a checklist or prescribed sequence; they are reasoning tools applied in context.

5. Security Is an Emergent Property

SP 800-160 addresses one of the most important ideas in trustworthy secure design: **security is an emergent property of an engineered system**. A system is expected to deliver required capabilities as authorized, as intended, and at the required performance level. It should not deliver unauthorized or unintended capabilities. Yet system-level behavior arises not only from the properties of individual components but also from how those components interact and compose. [4]

This distinction explains why a collection of individually secure components does not necessarily produce a trustworthy secure system. A cloud service may be correctly configured in isolation, an identity provider may authenticate correctly, a network gateway may enforce its rules correctly, and an AI agent may behave within its local policy—yet their composition can still create an unintended authority path, information flow, common-mode dependency, or failure condition.

SP 800-160 therefore requires systems engineers to address *emergence* at all levels of abstraction as systems are decomposed and recomposed. The security design principle of *Structured Decomposition and Composition* provides discipline for structuring the system; the principle of *Compositional Trustworthiness* asks whether the trustworthiness of system elements and their interactions is sufficient to support system-level claims. The principles of *Clear Abstractions* and *Reduced Complexity* improve the ability to reason effectively about those interactions. The principles of *Domain Separation* and *Least Sharing* together reduce unwanted coupling. And finally, the principle of *Mediated Access* makes necessary interactions explicit and controlled.

The AI era makes this concern more, not less, important. Agentic systems can introduce dynamic behavior, tool use, delegated authority, machine-speed interactions, and dependencies on external services. Even when each system component has an apparently reasonable local behavior, composition may create system-level outcomes no individual component was designed to produce. The engineering target must therefore remain the *authorized* and *intended* behavior of the system as a whole.

Trustworthy secure parts are necessary. Trustworthy secure composition is what determines whether the system behaves securely.

6. The Security Design Order of Precedence

The *Security Design Order of Precedence (SecDOP)* is the centerpiece of SP 800-160's argument about how systems engineers should control loss. Inspired by the NASA *System Safety Design Order of Precedence* [5], SecDOP seeks to minimize the loss potential built into the system's design. It establishes a secure structural context first and then introduces additional means of protection in *decreasing order of effectiveness*.

This order matters because cybersecurity practice often proceeds in the opposite direction. Organizations inherit an architecture, discover weaknesses, and then add security controls, monitoring, procedures, and training around it. SecDOP asks whether the susceptibility, hazard, or vulnerability can be removed or structurally reduced before compensating for it operationally. Table 1 summarizes the SecDOP alternatives, the engineering intent, and the associated security interpretations.

Table 1. SecDOP Alternatives, Engineering Intent, and Security Interpretations

SecDOP Alternatives	Engineering Intent	Security Interpretations
Eliminate through design selection	Remove susceptibility, hazard, or vulnerability so the loss path does not exist.	Eliminate unnecessary interfaces, authority, functions, dependencies, or exposed services.
Reduce through design alteration	Change the design to reduce frequency, severity, potential, or extent of loss.	Segment external-facing functions, separate domains, reduce privilege, restructure dependencies.
Incorporate engineered features or devices	Actively disrupt loss scenarios or reduce their effects.	Mediated access, trusted communications, redundant flows/elements, enforcement mechanisms.
Provide visibility and feedback	Detect hazardous/vulnerable conditions or actual loss and alert entities able to respond.	Anomaly detection, telemetry, protected logs, warnings, automated or human response.
Use signage, procedures, training, and equipment	Use administrative/operational means when stronger design alternatives are infeasible or inadequate.	Operating procedures, cautions, training, prescribed equipment; avoid relying on these alone.

Eliminate the Potential for Loss Through Design Selection

The strongest alternative is to select a design that removes the susceptibility, hazard, or vulnerability. SP 800-160 uses interface minimization as an example: every interface creates potential for susceptibility, hazard, and vulnerability, so a design with only the interfaces required to deliver the necessary functions has less inherent loss potential than one with unnecessary interfaces.

In modern terms, this can mean eliminating an unnecessary remote-management path, avoiding an architectural dependency that would collapse multiple trust domains into one, removing unused software or agent tools, avoiding unnecessary external connectivity, or selecting a component whose architecture does not require dangerous privileges. The security design principles of *Least Functionality*, *Least Sharing*, *Reduced Complexity*, *Minimal Trusted Elements*, and *Domain Separation* strongly reinforce this alternative.

Reduce the Potential for Loss Through Design Alteration

When the risky characteristic cannot be eliminated, the next choice is to alter the design to reduce the frequency, potential, severity, or extent of loss. The SP 800-160 example is *segmentation*: functionality that interacts with external entities can be separated from functionality that operates strictly within system boundaries.

This is the architectural heart of *Engineering for Compromise* [6]. A component may remain vulnerable, but the architecture can bound what that vulnerability can reach. The security design principles of *Domain Separation*, *Least Privilege*, *Least Sharing*, *Distributed Privilege*, *Hierarchical Protection*, *Defense in Depth*, and *Structured Decomposition and Composition* provide design reasoning for reducing propagation and blast radius.

Incorporate Engineered Features or Devices

Only after design selection and alteration are insufficient does SecDOP turn to engineered features and devices. SP 800-160 distinguishes mandatory features that apply foundational principles to interfaces, such as mediated access, from function-specific features such as redundant data or control flows and redundant system elements.

The important point is not that security controls are weak. It is that their effectiveness depends on structural context. A high-quality access control mechanism embedded in an architecture with excessive privilege, unnecessary connectivity, or bypass paths can still fail to prevent unacceptable loss. Conversely, a well-structured architecture gives mechanisms a tractable enforcement problem.

Provide Visibility and Feedback

When design and engineered features cannot adequately reduce loss potential, the system should provide timely detection and feedback to entities that are capable of responding. SP 800-160 identifies operational personnel, monitoring systems, and other systems as possible external responders. The design principle of *Anomaly Detection* is the most relevant, but the principles of *Commensurate Response*, *Continuous Protection*, and *Protective Recovery* also matter.

Visibility without response capability is not protection. Telemetry must arrive in time, with sufficient fidelity and context, and in a form the recipient can interpret. In AI-enabled environments this requirement may extend to machine-speed detection and automated containment, but the automation itself must be constrained and trustworthy.

Procedures and Training Are the Last Line of Defense, Not the Architecture

SecDOP places signage, procedures, training, and proper equipment last in priority. SP 800-160 explicitly cautions against using these as the only means to reduce loss potential. Human procedures remain necessary but not sufficient: they are vulnerable to time pressure, ambiguity, fatigue, turnover, social engineering, and adversarial manipulation.

This ordering is a useful corrective to security programs that respond to architectural weakness by writing another procedure or delivering another annual training module. If the system can be redesigned so the dangerous action is impossible, difficult, bounded, or automatically mediated, that is generally preferable to relying on perfect human behavior.

The best security control may be the attack path the architecture never created.

---

7. Functional Design Considerations

SecDOP establishes the order in which loss-control alternatives should be considered. Next, it is important to address the functional design considerations for trustworthy secure systems. These include: (1) assured functions that provide control enforcement, decision, and infrastructure; (2) essential mechanism design criteria; (3) security function failure analysis; (4) situational awareness; and (5) trade space considerations. Together, they turn architectural intent into implementable design discipline. [4]

Roles for Security-Relevant Control

SP 800-160 makes an important observation: all functions can influence behaviors and outcomes beyond themselves and therefore have *security relevance*. It nevertheless distinguishes three dedicated protection-control roles: (1) *decision functions*, (2) *enforcement functions*, and (3) *infrastructure functions*. Decision functions determine whether an action is authorized; enforcement functions impose the constraint; and infrastructure functions provide the trusted services and data on which decision and enforcement depend, such as secure storage, secure communications, and anomaly detection.

Ideally, other system functions should not interfere with these protection functions. If an unrelated function shares a privilege domain with a control function, for example, it may acquire elevated authority and create an avenue for attack. SP 800-160 therefore encourages architecture and allocation decisions that isolate critical control functions from unnecessary interference.

This reasoning maps naturally to the security design principles of *Minimal Trusted Elements*, *Least Privilege*, *Domain Separation*, *Least Sharing*, *Mediated Access*, and *Trustworthy System Control*. It also demonstrates why “zero trust” or access control cannot be reduced to simple identity checks. The trustworthiness of the decision, enforcement, and supporting infrastructure is itself part of the security problem.

Four Essential Design Criteria for Mechanisms

SP 800-160 states that protection mechanisms should be non-bypassable, evaluatable, always invoked, and tamper-proof [7]. These criteria generalize the *reference monitor concept* beyond a narrow access control implementation and provide a powerful test for any mechanism that claims to enforce a system constraint. Table 2 provides a summary of each design criterion and associated engineering questions.

Table 2. Security Design Criteria and Engineering Questions

Design Criterion	Engineering Question
Non-bypassable	Can any path, mode, maintenance function, exception, or alternate interface circumvent the mechanism?
Evaluatable	Is the mechanism sufficiently small, simple, and well-defined to support meaningful analysis and testing?
Always invoked	Is protection continuous across states, modes, failures, transitions, and operational exceptions?
Tamper-proof	Can the mechanism or the data on which it depends be modified without authorization?

The evaluatable criterion directly connects to the design principles of *Clear Abstractions*, *Reduced Complexity*, and *Structured Decomposition and Composition*. The always invoked criterion connects to the principle of *Continuous Protection*. Tamper resistance and non-bypassability connect to the principles of *Trustworthy System Control*, *Mediated Access*, *Domain Separation*, and *Minimal Trusted Elements*. Together, these criteria establish an architectural test: if a protection mechanism cannot be reasoned

about, can be bypassed, can be turned off at the wrong time, or can be altered by the thing it is supposed to constrain, confidence in the mechanism is fundamentally limited.

Security Function Failure Analysis

SP 800-160 requires explicit analysis of what happens when security functions fail. This follows directly from the security design principle of *Protective Failure*: failure of a system element should not itself create an unacceptable loss or trigger another loss scenario. Security functions are especially important because the system may depend on them to be continuously available and correct.

Failure analysis considers affected assets, loss consequences, allocation of functions to system elements, and interactions among function-element combinations. Crucially, this reasoning does not depend on predicting the exact event that causes failure. That makes it valuable against sophisticated or novel attack techniques. The analysis asks what the system can do when a trusted function is incorrect, compromised, absent, delayed, or unavailable.

The results should drive architecture, redundancy, privilege boundaries, recovery strategies, and the required rigor of *assurance*. A function whose failure can cause catastrophic mission loss should not receive the same engineering rigor as one whose failure causes a minor inconvenience.

Situational Awareness as a Designed Capability

SP 800-160 treats situational awareness as foundational. Access mediation may depend on timing, sequence, system state, or prior activity. Preventing and limiting loss requires data about system conditions. Situational awareness must therefore be designed to detect, capture, record, and analyze behavior at the frequency and granularity necessary for action.

The situational awareness mechanisms are themselves high-value targets. Logs, telemetry, state data, and attribution information can be altered to hide compromise or create false conclusions. Therefore, SP 800-160 applies the same essential design criteria to these mechanisms and suggests protections provided by applying the design principles of *Distributed Privilege* and *Domain Separation* for sensitive records.

This is especially important in AI-enabled operations. AI can accelerate analysis of telemetry, but a fast analytic engine cannot compensate for incomplete, manipulable, or untrustworthy source data. The trustworthiness of *observation* is part of the trustworthiness of *response*.

Trade Space: Security Must Compete Honestly

Trustworthy secure design does not assume unlimited resources. SP 800-160 explicitly places security in the engineering trade space. Asset value, criticality, exposure, and importance guide and inform the type, rigor, and expected effectiveness of protection. Costs include acquisition, development, integration, operation, sustainment, performance impact, services, documentation, training, and assurance.

An especially important point is that *assurance* has a cost. Two design options may provide similar nominal protection, yet one may be much easier to analyze and substantiate. In that case, the option with the lower assurance burden may be the better engineering choice. The security design principles of *Commensurate Protection*, *Commensurate Rigor*, and *Commensurate Trustworthiness* guide these decisions.

Trade-space analysis also guards against merely moving or transferring risk. A design change can reduce one exposure while creating another dependency or common-mode failure. System-level analysis is therefore essential; *local* optimization can produce *global* weakness.

8. Foundational Principles for Trustworthy Secure Design

SP 800-160 provides a set of 30 principles for reasoning about how a trustworthy secure design strategy can be realized [4]. The principles are not compliance rules and are not intended to be mechanically selected, prioritized, or sequenced independent of system context. Their value lies in helping engineers reason about the system in context. Several design principle clusters are particularly important to note as illustrated in Table 3 below:

Table 3. Trustworthy Secure Design Concerns, Principles, and Protection Contributions

Trustworthy Secure Design Concern	Applicable Design Principles	Protection Contribution
Reduce inherent loss potential	*Least Functionality*; *Least Sharing*; *Reduced Complexity*; *Domain Separation*; *Minimal Trusted Elements*	Remove unnecessary behavior, coupling, exposure, and trusted surface.
Control necessary interactions	*Mediated Access*; *Least Privilege*; *Distributed Privilege*; *Hierarchical Protection*	Constrain authority and make interactions explicit and enforceable.
Manage composition and emergence	*Structured Decomposition and Composition*; *Compositional Trustworthiness*; *Clear Abstractions*	Reason about system-level behavior arising from elements and interactions.
Maintain protection under adversity	*Defense in Depth*; *Redundancy*; *Diversity*; *Protective Defaults*; *Protective Failure*; *Protective Recovery*	Limit propagation and preserve or restore acceptable system states.
Know when assumptions fail	*Anomaly Detection*; *Continuous Protection*; *Loss Margins*; *Commensurate Response*	Detect deviations and respond before or after loss conditions arise.
Substantiate trust	*Commensurate Rigor*; *Commensurate Trustworthiness*; *Substantiated Trustworthiness*; *Trustworthy System Control*	Match evidence and engineering rigor to the consequences of failure.

The principles should therefore be seen as an engineering vocabulary for implementing the intent of trustworthy secure design. SecDOP says eliminate or reduce loss potential through design first; the principles help engineers determine what structural changes might accomplish that objective. Design considerations says mechanisms must be trustworthy; the principles help establish the properties and evidence needed to justify that trust.

These design decisions must also persist across the system life cycle. Trustworthy secure design is not applied only during the architecture and design definition life cycle processes. Its concepts are executed recursively and iteratively throughout the system life cycle—from mission analysis and stakeholder needs through requirements, architecture, design, implementation, integration, verification, validation, operation, maintenance, and disposal. *Architecture Definition* and *Design Definition* are where SecDOP most visibly shapes the solution, but changes in mission, technology, dependencies, or adversity can require the reasoning for trustworthy secure design to be revisited throughout the system life cycle.

9. From Good Design to Justified Confidence

A design can appear elegant and still be wrong. Trustworthiness therefore depends on *assurance*: grounds for justified confidence that claims have been or will be achieved [8]. That confidence comes from objective evidence that reduces uncertainty to an acceptable level. For trustworthy secure design, this means that claims such as “this interface cannot bypass mediation,” “this recovery path restores an authorized state,” “this security function is isolated from untrusted functions,” or “this architecture bounds compromise to one domain” require *evidence*. Verification and validation methods produce that evidence, and the evidence must be relevant, accurate, credible, and sufficient for reasoned judgment.

The connection to SecDOP is especially useful. Earlier design alternatives can sometimes produce simpler *assurance arguments*. Eliminating an unnecessary interface is generally easier to substantiate than proving that a complex control stack will safely mediate every possible use of that interface forever. The design principles of *Reduced Complexity* and *Minimal Trusted Elements* can therefore improve both security and the feasibility of assurance.

10. Applying Trustworthy Secure Design to the Frontier-AI Threat Space

Frontier AI should not cause organizations to abandon established systems security engineering. It should, however, cause them to apply it with greater urgency. AI-enabled cyber operations can increase discovery speed, automate portions of attack chains, scale social engineering and reconnaissance, assist exploit development, and increase the persistence with which exposed systems are probed. At the same time, defenders can use AI to discover vulnerabilities, analyze code, correlate telemetry, and accelerate their response.

Trustworthy secure design concepts and principles provide a technology-independent way to reason about this environment because it does not depend on knowing the adversary's exact tactics, techniques, or procedures. It asks what loss is unacceptable, what system states can produce that loss, what structural choices can eliminate or reduce those states, what mechanisms must enforce the remaining constraints, how those mechanisms can fail, what awareness is needed, and what evidence justifies trust.

Consider an AI-enabled mission-critical system with an agent that can call external tools. A control-centric approach may focus first on authentication, logging, filters, and model guardrails. A trustworthy secure design approach would ask earlier questions. Does the agent need each tool at all? Does it need direct network reachability? Can high-consequence tools be placed in a separate domain? Can authority be distributed so one agent cannot unilaterally cause a mission-critical action? Can the architecture make sensitive data unavailable to the agent unless a specific mediated transaction is authorized? If the agent or model is compromised, what system state results, and what loss remains possible?

That is SecDOP in practice: **eliminate unnecessary capability; alter the design to bound what remains; add trustworthy enforcement; provide visibility and response; and use procedures and training as supporting measures rather than the primary security architecture**.

The same reasoning applies to AI data centers, cloud platforms, critical infrastructure, space systems, defense systems, and enterprise environments. The technologies differ. The engineering questions remain remarkably stable.

11. A Practical Review for Engineering Teams

A systems engineering team can operationalize trustworthy secure design concepts and principles by using a small set of recurring review questions:

- What are the authorized and intended system behaviors and outcomes, including degraded and recovery modes?
- What system states, conditions, interactions, or transitions can produce unacceptable asset loss?
- Can the loss path be eliminated by selecting a different architecture, component, interface, dependency, or material?
- If it cannot be eliminated, can the architecture reduce its frequency, severity, propagation, or blast radius?
- Which engineered features and devices are then necessary, and are they non-bypassable, evaluatable, always invoked, and tamper-proof?
- Which functions make protection decisions, which enforce them, and which infrastructure functions do they trust?
- Can unrelated functions interfere with protection functions or inherit unnecessary privilege?
- What happens when each important security function is wrong, compromised, unavailable, delayed, or bypassed?
- What situational awareness is required to recognize approach to a loss condition and to support timely response?
- Do logs, telemetry, state data, and attribution mechanisms have protection commensurate with the trust placed in them?
- Have design alternatives been evaluated at the system level for cost, performance, assurance burden, and displaced risk?
- Which security design principles best help reason about the current design problem?
- Where in the system life cycle processes will each decision be implemented, verified, validated, monitored, and revisited?
- What evidence is required to justify the trust placed in the resulting architecture and mechanisms?

These questions are intentionally architecture-centered. They do not replace security control assessments, vulnerability scanning, penetration testing, red teaming, threat intelligence, or incident response. They *improve* and *strengthen* the system those activities are trying to protect.

12. Conclusion: Engineer the Loss Out Before You Manage It

NIST SP 800-160 contains a compact engineering philosophy with outsized relevance to today's cyber threat environment. It begins with *system behavior* rather than security controls. It treats security as an *emergent* property of an engineered system. It balances *preemptive* and *reactive* protection. It establishes an explicit order of precedence that favors eliminating and reducing loss potential through trustworthy secure design. It defines trustworthy roles and criteria for security mechanisms. It requires failure analysis and situational awareness. And it forces security into the same disciplined trade space as every other consequential system property.

The frontier-AI threat space makes this approach more urgent because defenders may have less time to compensate for weak architecture after deployment. Faster vulnerability discovery, automated attack chains, machine-speed exploitation, agentic behavior, and increasingly interconnected systems all increase the value of structural protection that does not depend on perfect prediction of the adversary.

The enduring lesson of the trustworthy secure design approach is therefore not "provide more security." It is more demanding: make better engineering decisions about where loss can arise and remove as much of that loss potential from the design as practicable before relying on mechanisms, monitoring, procedures, and people to manage what remains.

Do not begin by asking which security control to add. Begin by asking why the system permits the unacceptable loss to occur.

Acknowledgments

The author acknowledges the Institute for Security, Technology, and Society (ISTS) at Dartmouth College for supporting research and writing activities related to systems security engineering. The views expressed in this paper are those of the author and do not necessarily represent the views of Dartmouth College or the National Institute of Standards and Technology.

References

[1] Bank for International Settlements. A Mythos Moment? Frontier AI and Cyber Risk. BIS Bulletin No. 129, July 2026.

[2] Australian Signals Directorate / Australian Cyber Security Centre. Frontier AI Cyber Threat Considerations for Boards of Directors. August 2026.

[3] Frontier Model Forum. Managing Advanced Cyber Risks in Frontier AI Frameworks. February 2026.

[4] Ross, R., Winstead, M., and McEvilley, M. *Engineering Trustworthy Secure Systems*. NIST Special Publication 800-160, Volume 1, Revision 1. National Institute of Standards and Technology, November 2022. https://doi.org/10.6028/NIST.SP.800-160v1r1

[5] National Aeronautics and Space Administration (2011) System Safety Handbook Volume 1: System Safety Framework and Concepts for Implementation, NASA/SP-2010-580, Ver. 1.0. https://ntrs.nasa.gov/api/citations/20120003291/downloads/20120003291.pdf

[6] Ross, R. "Engineering for Compromise: Designing Systems That Remain Trustworthy, Secure, and Resilient Under Adversity." RONROSSECURE, LLC, LinkedIn, August 2026.

[7] Anderson J (1972) Computer Security Technology Planning Study, Technical Report ESD-TR-73- 51. Air Force Electronic Systems Division, Hanscom AFB. https://csrc.nist.gov/csrc/media/publications/conference-paper/1998/10/08/proceedings-of-the-21st-nissc-1998/documents/early-cs-papers/ande72a.pdf

[8] International Organization for Standardization/International Electrotechnical Commission/Institute of Electrical and Electronics Engineers (2019) ISO/IEC/IEEE 15026-1:2019, Systems and software engineering – Systems and software assurance – Part 1: Concepts and vocabulary. https://www.iso.org/standard/73567.html

