Engineering Trustworthy Secure AI Infrastructure

How NIST SP 800-160 Design Principles and Life Cycle Processes Make RAND's Secure Inference Data Center Trustworthy and Resilient

Dr. Ron Ross

CEO, RONROSSECURE, LLC.

Abstract

RAND's report, Secure Inference Data Centers: A Vertically Integrated Strategy for Security Engineering, proposes a purpose-built architecture for protecting strategically important artificial intelligence models and inference data against an exceptionally capable nation-state adversary. Its strongest contribution is not a collection of controls but a vertically integrated security argument: unacceptable losses drive hazards, constraints, requirements, architecture, formally verified boundary mechanisms, and circuit-level implementations. This paper argues that NIST Special Publication 800-160 provides the design foundation for that argument and the engineering discipline to sustain it across the entire system life cycle. The relationship is synergistic. NIST provides the systems security engineering fundamental principles, concepts, and life cycle processes; RAND demonstrates how selected principles can be composed into an emergent secure inference capability. Domain separation, minimality, mediated access, protective defaults, protective failure, detection, recovery, and evidence operate as a causal chain in which unauthorized behavior is progressively eliminated, constrained, detected, contained, and recovered from. The system life cycle processes ensure that this chain begins with mission and stakeholder protection needs, is preserved through requirements, architecture, design, implementation, integration, verification, transition, and validation, and remains effective during operation, maintenance, and disposal. The resulting SIDC security case is not an operational overlay. It is a controlled body of traceable claims, assumptions, decisions, and objective evidence that evolves with the system from conception to retirement.

1. Introduction

Artificial intelligence has become a strategic source of national power, economic competitiveness, military advantage, scientific discovery, and critical-infrastructure capability. The security of the United States increasingly depends on its ability to protect critical AI intellectual property—including models, algorithms, training and reference data, system architectures, AI-enabled products, operational services, and the knowledge required to reproduce or subvert them. Nation-state adversaries are not limited to stealing these assets. They may seek to manipulate models, corrupt inference, compromise supporting systems, insert latent access paths, undermine confidence in U.S. AI products and services, or acquire capabilities that offset American technological advantage.

Conventional cybersecurity is necessary but may be insufficient for the most consequential AI assets. General-purpose cloud and enterprise environments are optimized for connectivity, scale, rapid change, resource sharing, and availability. Those characteristics can conflict with the containment, minimality, predictability, and high assurance required when an adversary has the resources, access, expertise, patience, and operational discipline of a nation-state. Protecting the highest-value AI assets therefore requires an engineering strategy in which security is a governing system property rather than an overlay of controls.

RAND's Secure Inference Data Center (SIDC) provides a compelling architectural response: a compact, purpose-built cyber-physical facility that sharply limits information flows, separates trust domains, minimizes functionality and privilege, and carries selected security constraints from system losses to formally analyzed boundary mechanisms and hardware-realizable implementations [1]. When grounded in the systems security engineering principles and life-cycle processes of NIST SP 800-160 Volume 1 Revision 1, the SIDC can be understood as the "Fort Knox" of digital asset protection; a facility whose architecture, operations, maintenance, recovery, and retirement are all organized around preserving the confidentiality and integrity of critical AI assets against nation-state subversion.

The Fort Knox analogy is intentionally demanding. A secure vault is not defined solely by the strength of its door. Its trustworthiness depends on the site, structure, access rules, personnel, custody procedures, monitoring, response, maintenance, and continuing evidence that every part of the protection system performs as intended. The same is true for the SIDC. RAND supplies a high-assurance architecture and a vertically integrated security methodology. NIST SP 800-160 supplies the design principles that make the architecture coherent and the system life cycle processes that preserve its security meaning from mission analysis through disposal. The purpose of this article is to combine those contributions into a *holistic engineering argument* for protecting the nation's most critical AI intellectual property.

The remainder of the article develops the argument in seven sections:

- **Section 2 — Establish the RAND contribution and engineering challenge.** Explain the SIDC threat context, core architectural choices, concept-to-circuit method, and the continuity-of-intent problem that remains between a strong design and a trustworthy operational capability.
- **Section 3 — Connect NIST SP 800-160 security design principles to RAND's architecture.** Show how composed NIST design principles provide the foundation for RAND's five emergent SIDC system properties and progressive loss control.
- **Section 4 — Define the operational obligations.** Translate the principle composition into six non-overlapping obligations for boundary integrity, bounded compromise, mediated transactions, protective states, trusted setup and recovery, and continuing assurance and change control.
- **Section 5 — Apply security across all NIST SP 800-160 system life cycle processes.** Demonstrate how security intent is generated, transformed, realized, verified, fielded, sustained, and retired across the complete system life cycle, not merely protected during operation.
- **Section 6 — Provide a repeatable engineering method.** Convert the design principles and life-cycle argument into seven practical steps that can serve as acquisition, engineering-review, authorization, operational, and change-control decision gates.
- **Section 7 — Identify how NIST SP 800-160 extends the RAND work.** Address life-cycle completeness, explicit composition, and evidence governance as the disciplines that are needed to close RAND's acknowledged implementation backlog without weakening its architecture.
- **Section 8 — State the holistic conclusion.** Define the decisive test for SIDC trustworthiness: every function, flow, privilege, transition, exception, change, recovery action, and disposal decision remains necessary, constrained, traceable, and supported by evidence commensurate with the losses at stake.

2. The RAND Contribution and the Remaining Engineering Challenge

RAND's report addresses a difficult and increasingly important problem: some AI inference deployments may require protection beyond what ordinary commercial cloud environments can credibly provide. The proposed secure inference data center (SIDC) is a compact, purpose-built cyber-physical facility intended to preserve the confidentiality and integrity of AI intellectual property (IP) including model weights, prompts,

algorithms, reference data, responses, and inference computation against an Operational Capacity 5 adversary[1] over a bounded operational period. RAND deliberately accepts major constraints on availability, connectivity, throughput, software updates, and convenience to achieve that objective [1].

The report's key methodological contribution is its *concept-to-circuit* approach.[2] The System-Theoretic Process Analysis for Security (STPA-Sec)[3] begins with goals and unacceptable losses, derives hazards and constraints, and carries those constraints into system and component requirements. At security-critical boundaries, formal protocol analysis, model checking, hardware realization, circuit assertions, testing, and red-team evaluation create a layered assurance chain [1]. This preserves the reason a mechanism exists, not merely its technical specification.[4]

That approach is strongly aligned with NIST SP 800-160. RAND explicitly invokes the publication's Security Design Order of Precedence and its problem, solution, and trustworthiness contexts [1]. SP 800-160 adds something even more important for implementation: a coherent set of design principles that constrain how the system is decomposed, how authority and information flow are controlled, how failures and recovery are handled, and what evidence is required before trust is granted [2].

The operational challenge is therefore continuity of intent. The SIDC's claims can fail without any obvious violation of its high-level architecture. A maintenance procedure can create a temporary bypass. A recovery path can restore compromised state. A new monitoring function can become an exfiltration channel. A formally verified component can be integrated under assumptions that no longer hold. An operator can accumulate privileges across realms. SP 800-160 provides the engineering discipline for preventing those implementation and life-cycle gaps.

3. NIST SP 800-160 as the Foundation for RAND's Architecture

NIST SP 800-160, Volume 1, Revision 1 Appendix E is best understood as a set of interdependent security design principles that are applied throughout the system life cycle. RAND's architecture demonstrates the resulting causal chain: The principle of *Domain Separation* establishes realms; the principles of *Least Sharing*, *Least Privilege*, and *Least Functionality* reduce coupling and attack surface; the principle of *Mediated Access* controls the interactions that remain; the principles of *Protective Defaults* and *Protective Failure* constrain deviations; the principle of *Anomaly Detection* identifies degradation or prohibited behavior; the principle of *Protective Recovery* restores an authenticated state; and the principles of *Substantiated and Compositional Trustworthiness* provide evidence that the mechanisms and their integration preserve system-level constraints [2].

This composition explains RAND's five SIDC properties. The properties are emergent outcomes of the architecture, including cyber, physical, procedural, and human elements, not features supplied by a single control. The mapping below identifies the primary dependencies without implying that other principles are irrelevant.

---

[1] Operational Capacity 5 (OC5) is RAND's highest adversary-capability category for the SIDC analysis. It describes a nation-state-level adversary able to coordinate sophisticated cyber, physical, supply-chain, intelligence, and insider operations over time.

[2] Concept-to-circuit is RAND's term for maintaining traceability from system-level losses and constraints through architecture, protocol specifications, formal models, hardware realization, circuit assertions, and verification evidence.

[3] STPA-Sec applies system-theoretic hazard analysis to security. It begins with unacceptable losses and identifies hazardous system states, unsafe control actions, and enforceable security constraints rather than beginning with a catalog of threats or controls.

[4] A realm is an intentionally separated trust and control domain with defined resources, privileges, interfaces, and information-flow rules. RAND uses realms to prevent compromise in one portion of the SIDC from automatically conferring authority or access in another.

Table 1. RAND system properties as emergent results of composed NIST SP 800-160 design principles.

RAND System Property	NIST SP 800-160 Design Principle Composition	Operational Effect
P1: Cross-boundary flow containment	Domain Separation; Least Sharing; Mediated Access; Protective Defaults; Trustworthy System Control	Only enumerated flows can cross established boundaries, and every permitted crossing is mediated and non-bypassable.
P2: Computational-state confidentiality	Domain Separation; Least Privilege; Least Functionality; Minimize Detectability; Continuous Protection	Plaintext state remains inside its originating realm and unnecessary exposure paths do not exist.
P3: Integrity of inference	Hierarchical Protection; Mediated Access; Substantiated and Compositional Trustworthiness; Trustworthy System Control	Authenticated artifacts, controlled computation, reference baselines, and evidence preserve the inference chain.
P4: Assured recovery	Anomaly Detection; Protective Failure; Protective Recovery; Redundancy; Self-Reliant Trustworthiness	Hazards are detected, contained, and restored from an authenticated basis without creating another loss.
P5: Cross-realm privilege separation	Domain Separation; Least Privilege; Least Sharing; Distributed Privilege; Structured Decomposition	Compromise of one realm, component, or person does not confer unauthorized authority in another realm.

The architectural significance is progressive loss control. Unnecessary behavior is eliminated; necessary behavior is bounded and mediated; deviations move toward containment; residual hazards are detected; recovery returns the system to an authorized state; and assurance evidence supports the resulting claims. Section 4 addresses what must be done to preserve that chain in a fielded SIDC rather than repeating the definitions of the principles.

4. Operational Obligations for a Trustworthy SIDC

Section 3 established how the NIST SP 800-160 security design principles combine to produce RAND's five SIDC properties. This section translates that design logic into six operational obligations that must remain effective across normal service, exceptional access, degradation, failure, recovery, maintenance, and change. The obligations are intentionally complementary: each addresses a different way in which a sound architecture can lose its security meaning after implementation. Together, these obligations form the operational bridge between RAND's architecture and the life-cycle processes examined in Section 5. They ensure that the SIDC does not merely begin in a secure configuration, but continuously preserves its realm boundaries, authorized behavior, recovery basis, and evidentiary foundation throughout its service life.

Preserve Boundary Integrity in Every System State

RAND's realm partitioning, fixed diode-gated topology, and verified cross-boundary protocols establish the SIDC's core invariants. Their operational meaning is broader than inference traffic. Installation, diagnostics, calibration, emergency access, maintenance tooling, removable media, electromagnetic emissions, waste, power, cooling, personnel movement, and recovery must all conform to the same realm and flow rules. A boundary that is non-bypassable during normal service but bypassable during commissioning or repair is not trustworthy.

Each temporary or exceptional connection must therefore be modeled as a cross-realm interaction, justified by a requirement, assigned an owner, constrained in authority and duration, logged, tested, and removed or disabled after use. Verification must address indirect coupling through shared writable storage, management services, utilities, time sources, personnel, and diagnostic equipment, not merely visible network paths.

Make Bounded Compromise and Minimality Enforceable

RAND assumes that commodity elements and individual subsystems may be compromised. The system claim survives only when compromise remains local. Privileges must be modeled across cyber, physical, and procedural domains; separation-of-duty rules must resist the stated insider threshold; emergency authority must be time-bounded and observable; and no maintenance, recovery, or administrative role may accumulate a cross-realm capability that the normal architecture prohibits.

Minimality must be an enforced configuration property, not an operator's intention. Approved baselines should specify permitted hardware, firmware, software, protocols, services, interfaces, persistence, and privilege. Disallowed capability should be absent, physically constrained, cryptographically disabled, or continuously attested. The decisive assessment question is whether any reachable combination of roles, credentials, tools, shared services, or latent functions can create an unauthorized cross-realm capability.

Define Every Necessary Crossing as a Bounded Transaction

Every permitted boundary interaction requires a precise transaction model: message or transfer vocabulary, authenticated endpoints, maximum capacity, timing behavior, privilege conditions, audit semantics, failure response, and decommissioning rule. Unknown, malformed, mistimed, excessive, or unauthenticated activity must map to a defined hazard response rather than permissive error handling. Test ports, calibration links, consoles, replacement hardware, key ceremonies, facility controls, and emergency procedures must not become alternate routes around the mediator.

Formal protocol analysis can establish selected properties of the digital transaction model, while physical and procedural analysis must address channels that software access control cannot see. The complete-mediation claim is credible only when all crossing mechanisms and exception paths are included in the system boundary.

Engineer Detection, Failure, and Recovery as One Protective State Model

The SIDC needs a closed loss-control loop rather than an assumption of comprehensive prevention. Monitoring should be derived from invariants: engineers specify which observations indicate violation or degradation, required detection time and confidence, and the response authority. When authentication, timing, telemetry, energy flow, sensors, or controllers depart from valid assumptions, the system should move toward a predefined protective state before confidentiality or integrity is sacrificed.

Containment, degraded service, investigation, authenticated restoration, reconstitution, and deliberate destruction require explicit state transitions. For each hazard, the design must identify which flows cease, which authorities remain valid, what evidence is preserved, and what conditions permit exit. Recovery includes hardware identity, keys, reference data, physical boundaries, measurement tools, and operator authority, not only a known-good software image. Exercises must demonstrate restoration without disclosure, rollback to a vulnerable state, uncontrolled privilege expansion, or destruction of needed evidence.

Treat Trusted Setup and Recovery as Security-Critical Systems

RAND acknowledges that the SIDC cannot detect pre-ingestion compromise when trusted setup is itself compromised [1]. Trusted setup, therefore, requires its own protection needs, architecture, separation of duties, controlled ingestion paths, provenance, verification, and signed evidence package. That package should bind hardware, keys, baseline measurements, model weights, reference data, results, facility configuration, procedures, personnel roles, tools, exceptions, and custody records into the authorized configuration.

Recovery requires comparable protection because corruption of the recovery basis can turn a detectable incident into recurring compromise. Foundational changes (e.g., hardware replacement, root-key change, model replacement, boundary modification, or recovery-vault alteration) must trigger a defined degree of reestablishment, reverification, and reauthorization.

Govern Assurance Evidence and Change as Continuing System Functions

Formal proofs support specified properties under stated assumptions; they do not prove that an entire facility is secure [1]. Each unacceptable loss must trace through hazards, constraints, requirements, architecture, component claims, implementation and integration evidence, verification, operating procedures, and monitors. Component evidence is sufficient only when environmental assumptions and dependencies are satisfied and composition preserves the system constraint. The assurance case must be versioned, protected, and bound to the active configuration.

Change must be constrained rather than prohibited categorically. Every proposed change should reenter the loss-driven pipeline: identify the need; determine affected losses, hazards, assumptions, and claims; choose the minimum mechanism; analyze new flows and dependencies; implement and integrate under control; produce evidence commensurate with significance; validate in a representative environment; and reauthorize affected claims. Emergency change may use a pre-engineered narrower path, but it cannot waive impact analysis, monitoring, or post-action review.

5. Engineering the SIDC Across the NIST SP 800-160 Life Cycle Processes

NIST SP 800-160 Volume 1, Revision 1 explains how a trustworthy secure system is developed and how the security intent is generated, transformed, realized, demonstrated, used, sustained, and retired. NIST uses system life cycle processes from ISO/IEC/IEEE 15288 and adds security purposes, outcomes, activities, and tasks to each [2]. The processes are iterative and recursive rather than a simple waterfall. They may be applied at multiple system levels, revisited when assumptions or configurations change, and used in different life-cycle stages. For an SIDC, that means the same principles that constrain a diode controller also constrain the facility architecture, trusted-setup system, operating procedures, maintenance infrastructure, recovery capability, and disposal method.

This process view corrects a potentially dangerous interpretation of the RAND proposal. Operational discipline cannot compensate for protection needs omitted during mission analysis, an under-specified boundary, an architectural dependency that defeats realm separation, an implementation that diverges from its model, or an integration environment that violates proof assumptions. Conversely, a strong initial design can decay if transition, maintenance, and disposal are treated as administrative matters. Security must be engineered into the work products and decision criteria of every life-cycle process.

Business or Mission Analysis: Establish the Security Purpose and Claim Boundary

The first life-cycle process, *Business or Mission Analysis*, defines the strategic problem or opportunity, characterizes the solution space, and selects feasible solution classes. For the SIDC, this process must state

why ordinary cloud or enterprise architectures cannot satisfy the protection objective against the assumed adversary, which mission consequences justify a purpose-built facility, and what trade space decisions are acceptable. The RAND choices to bound the operating period, privilege confidentiality and integrity over availability, restrict connectivity, and reduce throughput are mission decisions before they are technical decisions. The SP 800-160 security design principles including *Commensurate Protection*, *Commensurate Rigor*, *Commensurate Trustworthiness*, and *Loss Margins* ensure that those choices correspond to the consequences and uncertainty at stake.

The principal outputs should include a security-relevant mission problem statement; candidate solution classes; a preliminary concept of secure operations; threat, adversity, and uncertainty assumptions; loss tolerances; and a defined claim boundary. Alternatives should include different technical architectures and different mission concepts, such as reducing the sensitivity of workloads, distributing the capability, limiting retention, or changing how results are consumed. The selected SIDC solution class must trace back to the losses it is intended to prevent. If the adversary, mission duration, workload, or accepted availability trade changes, the mission analysis and downstream claims must be revisited.

Stakeholder Needs and Requirements Definition: Convert Consequences into Protection Needs

The *Stakeholder Needs and Requirements Definition* process identifies security-relevant stakeholders, assets, adversities, loss concerns, protection priorities, constraints, measures of effectiveness, and stakeholder requirements. SIDC stakeholders extend beyond model owners and operators. They include inference customers, data providers, facility and safety personnel, maintainers, incident responders, authorizing officials, intelligence and counterintelligence organizations, supply-chain organizations, regulators, and communities affected by physical operations. Their interests may conflict: a customer may demand rapid output, an investigator may require evidence retention, and the security authority may require shutdown or destruction.

SP 800-160 design principles provide engineering discipline to this negotiation. The *Commensurate Protection* principle ties protection needs to consequences; the principle of *Domain Separation* identifies which stakeholders and assets require independent realms; the *Least Sharing* principle challenges demands for common services; the principle of *Protective Failure* makes explicit when availability may be sacrificed; and *Commensurate Response* principle identifies authorized reactions to hazards. Representative scenarios must cover normal inference, setup, degraded operation, emergency response, maintenance, recovery, and disposal. The output is not a list of controls but an agreed, prioritized set of stakeholder protection needs and measurable success criteria, with bidirectional traceability to the people, assets, losses, and scenarios from which they arose.

System Requirements Definition: Express Security as Complete, Testable System Obligations

The *System Requirements Definition* process transforms the stakeholder view into a technical view. For the SIDC, RAND's unacceptable losses and STPA-Sec constraints become requirements for all system states, modes, transitions, interfaces, functions, and boundaries. Requirements should define the five RAND system properties, the realm model, allowed and prohibited flows, privilege limits, trusted-setup integrity, recovery conditions, audit semantics, response times, and assurance expectations. They must also address non-cyber channels (i.e., people, power, cooling, electromagnetic emanations, removable media, waste, and maintenance equipment), because a facility-level confidentiality claim cannot stop at a network interface.

The SP 800-160 design principles improve requirement quality. The principle of *Protective Defaults* becomes a requirement that uninitialized, ambiguous, or unauthorized transactions are denied. The principle of *Continuous Protection* requires that security properties hold through startup, shutdown, degradation, maintenance, and recovery. The *Mediated Access* principle requires complete mediation of

every allowed boundary interaction. The principle of *Minimal Trusted Elements* creates constraints on the size and function of the trusted mechanisms. Each requirement needs rationale, security metadata, criticality, verification method, applicable states, assumptions, and links upward to protection needs and downward to architecture and evidence. Requirements analysis must expose conflict (e.g., whether synchronous audit latency or recovery objectives create a pressure to bypass containment).

System Architecture Definition: Allocate Trust, Authority, and Information Flow

The *Architecture Definition* process establishes the fundamental properties of the system and allocates requirements to system elements. This is the process in which the RAND realm architecture becomes authoritative rather than illustrative. The design principles of *Domain Separation* and *Structured Decomposition and Composition* establish realms and their responsibilities; the *Hierarchical Protection* principle orders dependencies; the principles of *Least Sharing* and *Least Privilege* constrain the shared resources and authority; the principle of *Mediated Access* defines permitted interfaces; and the principle of *Trustworthy System Control* identifies non-bypassable control points. The physical facility, workforce, trusted setup, recovery vault, monitoring plane, utilities, logistics, and support systems belong in the architecture alongside computing and communications capabilities.

Architecture evaluation should examine alternative decompositions and attack paths, common-mode dependencies, covert or unintended channels, failure propagation, insider combinations, and the satisfaction of component assumptions by surrounding elements. An architecture description should include behavioral views for normal and exceptional states, data and control-flow views, trust and privilege views, physical and cyber-physical views, and assurance views that show where evidence will be obtained. The result should demonstrate why the chosen structure makes unacceptable losses difficult to reach and why remaining critical mechanisms are small enough to analyze rigorously.

Design Definition: Turn Architectural Intent into Buildable Specifications

The *Design Definition* process refines the architecture into detailed realizations for system elements. At this level, a cross-realm controller requires precise message formats, state machines, timing rules, privilege checks, reset behavior, error handling, audit events, physical interfaces, initialization procedures, and prohibited states. Facility boundaries require equally detailed specifications for doors, conduits, removable media, maintenance ports, waste paths, utilities, and personnel workflows. The *Clear Abstractions* principle preserves the difference between security intent and mechanism; the principles of *Least Functionality* and *Reduced Complexity* remove unnecessary behavior; the principles of *Protective Defaults* and *Protective Failure* define behavior under uncertainty and fault; and the principles of *Diversity*, *Redundancy*, and *Distributed Privilege* are applied where common-mode or single-point compromise is unacceptable.

Every selected design characteristic should trace to a requirement and include an explicit security rationale. The design must identify trusted elements, dependencies, environmental assumptions, implementation constraints, and evidence needs. Trade studies should compare not only cost and performance but also attack surface, state-space size, ease of verification, recovery behavior, supply-chain exposure, and the likelihood that maintainers will need exceptional access. A design that is secure only when operated perfectly is not an adequate design for an OC5 environment.

System Analysis: Maintain a Rigorous Basis for Security Decisions

The *System Analysis* process provides the quantitative and qualitative basis for decisions throughout the system life cycle. For the SIDC, it includes STPA-Sec hazard analysis, adversary and uncertainty analysis, information-flow and privilege analysis, covert-channel assessment, formal modeling, reliability and common-cause analysis, physical security analysis, human-error and insider-path analysis, performance-

security trade studies, and analysis of recovery and destruction options. It should be used whenever alternatives, changes, anomalies, or conflicting requirements require an evidence-based decision.

The design principles determine the questions analysis must answer. The *Loss Margins* principle asks how close the system may approach a hazardous state. The *Commensurate Rigor* principle calibrates analytical depth to consequence. The principle of *Compositional Trustworthiness* tests whether local claims survive interaction. The *Self-Reliant Trustworthiness* principle identifies external dependencies that can invalidate the claim. System analysis results including uncertainties, model limitations, assumptions, and dissenting conclusions must be controlled and traceable. The objective is not to produce more models; it is to prevent security-significant choices from becoming unsupported judgments.

Implementation: Realize Elements Without Losing the Security Meaning

The *Implementation* process realizes specified hardware, software, firmware, physical structures, procedures, training, and data. Concept-to-circuit is strongest here: formal protocol models and circuit assertions can constrain the realized cross-boundary mechanism. But implementation assurance must extend to compilers, synthesis tools, libraries, hardware descriptions, manufacturing, firmware loading, facility construction, installation scripts, key ceremonies, baseline data, operator procedures, and acceptance of supplied elements. The principle of *Minimal Trusted Elements* limits what must be trusted; the principle of *Reduced Complexity* makes rigorous inspection more practicable; the principle of *Least Functionality* prevents latent capability; and the principle of *Substantiated Trustworthiness* requires objective evidence rather than pedigree or reputation.

Implementation records should preserve provenance, configuration identifiers, tool versions, build parameters, inspection results, deviations, vulnerabilities, and the relationship between the realized element and its approved specification. Reproducible builds, signed artifacts, controlled manufacturing data, destructive or nondestructive inspection, and independent review may be justified for the most critical elements. Any implementation deviation must undergo impact analysis before acceptance; otherwise, the implementation becomes a silent break in the security argument.

Integration: Demonstrate That Secure Elements Compose Securely

The *Integration* process combines realized elements into larger assemblies and ultimately the complete facility. This is where individually credible components can produce insecure emergent behavior. The SIDC integration strategy should therefore follow the architecture's trust hierarchy and verify assumptions at each composition boundary. Integration must address cyber, physical, procedural, and human interfaces; initialization order; key and identity binding; maintenance connections; time synchronization; error propagation; logging; degraded modes; and removal of temporary test equipment. The principle of *Compositional Trustworthiness* is the controlling principle, supported by the principles of *Mediated Access*, *Hierarchical Protection*, *Protective Defaults*, and *Trustworthy System Control*.

Integration evidence should show that only specified interfaces exist, that prohibited paths remain absent, that component assumptions are met, and that system properties survive representative faults and adversarial actions. Temporary integration mechanisms must be treated as capabilities subject to least privilege and least persistence. The integrated baseline should bind physical serial numbers, firmware and software measurements, facility configuration, keys, models, reference data, procedures, and personnel authorities into one identifiable configuration. Without that binding, later verification and authorization evidence cannot be reliably associated with the fielded SIDC.

Verification: Prove the System Was Built Right

The *Verification* process provides objective evidence that the system and its composed elements satisfy specified requirements and design characteristics. The SIDC verification program should use multiple, complementary methods: inspection, analysis, formal proof, model checking, static analysis, simulation, test, fault injection, adversarial evaluation, and demonstration. Formal methods are especially valuable for minimal, deterministic boundary controllers, but the claim must remain bounded to the properties modeled and the assumptions stated. Facility flows, personnel actions, recovery procedures, and cyber-physical interactions require other forms of evidence.

A *requirements-verification matrix* should identify the method, environment, configuration, expected result, independence level, and acceptance authority for every security requirement. The principle of *Continuous Protection* requires verification across states and transitions, not just steady-state operation. The principles of *Protective Failure* and *Protective Recovery* require tests of ambiguous inputs, partial failures, loss of telemetry, corrupted recovery data, power interruption, and adversarial attempts to exploit the response. Failures and waivers must link back to affected claims, hazards, and risks; they cannot be closed as isolated test discrepancies.

Transition: Establish the Authorized Capability in Its Actual Environment

The *Transition* process moves the verified system into its operational environment and establishes the capability to deliver services. For an SIDC, transition includes secure transport and installation, facility commissioning, environmental characterization, key and identity establishment, trusted ingestion of model and reference data, personnel qualification, procedure validation, creation of the recovery basis, removal of construction and test access, and authorization to enter the operational state. Trusted setup is therefore a transition system with its own architecture, protection needs, controlled configuration, and evidence, not a pre-operational checklist.

Transition acceptance must confirm that the operational environment satisfies assumptions used in analysis and verification. It should reconcile the as-built and as-designed baselines, close temporary paths, establish monitoring and response readiness, inventory residual risks, and seal a signed setup evidence package. The principle of *Protective Defaults* should keep the facility non-operational until the required evidence and approvals are present. If the deployment environment differs materially from the verified environment, the affected requirements, analyses, tests, and authorization claims must be renewed before service begins.

Validation: Prove the Right Secure Capability Exists for the Mission

The *Validation* process asks whether the system, when used in the intended operational environment, satisfies stakeholder needs and intended use (i.e., is fit for purpose). Verification can show that a controller implements its protocol correctly while validation reveals that the protocol, facility workflow, or mission concept is inadequate. SIDC validation should use end-to-end mission scenarios involving realistic users, inference workloads, adversary actions, degraded conditions, insider combinations, emergency response, recovery, and mission trade-offs. It should test whether confidentiality and integrity are preserved without making the capability unusable in ways stakeholders did not accept.

Validation evidence should address measures of effectiveness established during stakeholder analysis: whether protected inference can be delivered, whether prohibited flows remain unavailable, whether detection and response are timely enough to prevent loss, whether operators can execute protective procedures under stress, and whether recovery returns the system to an authorized state. Stakeholder acceptance must be informed by residual risks, unresolved assumptions, and operational limitations. Validation is repeated when mission use, stakeholders, the environment, or major system behavior changes.

Operation: Deliver Services While Preserving the Authorized Security Case

The *Operation* process uses the SIDC to deliver defined services. The earlier sections of this paper address many operational mechanisms, but the life-cycle view changes their role: operators execute an architecture and evidence-based security case established upstream. Operational procedures must preserve the realm model, enforce privilege and separation of duties, control workload and data ingestion, monitor invariants, manage anomalies, contain hazards, preserve evidence, and authorize recovery. The *Continuous Protection* principle requires those rules be enforced across every operating mode, including startup, idle, output release, degraded service, emergency containment, investigation, restoration, and shutdown.

Operational data should be used to test the continuing validity of assumptions and measures, not merely to detect known attacks. Changes in workload, latency, component behavior, personnel patterns, physical environment, threat intelligence, or false-positive rates may signal that a requirement or design assumption no longer holds. The assurance case must therefore be a living operational artifact linked to the active configuration. Operations can identify the need for change, but it cannot bypass the other system life cycle processes to introduce that change.

Maintenance: Sustain Capability Without Creating a Parallel Attack Architecture

The *Maintenance* process sustains the system's capability and includes corrective, preventive, adaptive, and perfective actions. It is a high-risk SIDC process because diagnostics, replacement parts, vendor tools, privileged personnel, removable media, and temporary connectivity can recreate the very cross-realm capabilities the architecture eliminated. Maintenance must therefore have an explicit secure concept, isolated support environments, authenticated tools and parts, controlled entry and exit paths, separation of duties, time-bounded privileges, verified restoration, and complete removal of temporary mechanisms. The principles of *Least Persistence*, *Least Privilege*, *Mediated Access*, and *Protective Defaults* are especially important.

Every maintenance action should begin with an impact analysis and end with configuration reconciliation and renewed evidence proportional to the change. Replacement of a critical component may require provenance review, re-binding of identity and keys, regression verification, recovery-baseline update, and partial reauthorization. Fixed software may reduce update-channel risk, but it cannot justify preserving a known vulnerable state. A pre-engineered change process, using the same requirements, architecture, design, implementation, integration, verification, transition, and validation logic at an appropriate scale, allows the SIDC to evolve without normalizing uncontrolled access.

Disposal: End the System Without Releasing Its Residual Security Value

The *Disposal* process ends the existence of a system or element, but confidentiality, integrity, legal, safety, and intelligence consequences may persist. An SIDC can retain model weights, prompts, outputs, keys, firmware, design details, audit records, physical evidence, and architectural knowledge after operational shutdown. Disposal requirements must therefore be established early and designed into media, hardware, storage, key management, facility construction, contracts, records, and logistics. The *Least Persistence* principle minimizes residual sensitive state; the principles of *Protective Recovery* and *Protective Failure* constrain final transitions; the principle of *Mediated Access* controls custody; and the *Substantiated Trustworthiness* principle requires evidence that sanitization, destruction, transfer, or retention was accomplished as intended.

A disposal plan should define authorization, chain of custody, sanitization and/or destruction methods, verification, witness requirements, environmental and safety constraints, disposition of records, handling of reusable components, termination of credentials and trust relationships, and preservation of evidence required for accountability. Decommissioning must also update any dependent systems and invalidate

assumptions that relied on the SIDC. The security claim does not end when power is removed but when residual assets and dependencies have reached approved terminal states supported by objective evidence.

A Life-Cycle Assurance Thread for the SIDC

The system life cycle processes should produce one connected assurance thread rather than multiple threads. Each process consumes security meaning from earlier work, adds decisions and evidence, and exposes conditions that may force upstream revision. The table identifies the minimum security question and representative evidence for each system life cycle process. It is a governance aid, not a substitute for tailoring the detailed life cycle activities and tasks.

Table 2. The continuous SIDC assurance thread across SP 800-160 system life cycle processes.

Life Cycle Process	Controlling SIDC Security Question	Representative Evidence or Baseline
Business or Mission Analysis	Why is an SIDC the justified solution class?	Mission security problem; alternatives and trade study; claim boundary; adversary and loss assumptions
Stakeholder Needs and Requirements	Whose assets, losses, constraints, and success measures govern?	Stakeholder map; asset/loss analysis; scenarios; prioritized protection needs; agreement
System Requirements Definition	What must the complete system do and never do in every state?	Security requirements baseline; rationale and metadata; state/mode rules; verification criteria
System Architecture Definition	How are trust, authority, information flow, and failure contained?	Realm, flow, privilege, physical, behavioral, and assurance views; architecture evaluation
Design Definition	What precise mechanisms and procedures realize the architecture?	Detailed specifications; state machines; interface rules; design trade studies; trusted-element list
System Analysis	What evidence supports consequential decisions and residual risk judgments?	Hazard, threat, formal, reliability, human, physical, and trade-space analyses
Implementation	Does each realized element faithfully embody its approved design?	Provenance; build and construction records; inspections; configuration identifiers; deviations
Integration	Do secure elements compose without new paths or invalid assumptions?	Integration plan/results; interface inventory; assumption checks; integrated baseline
Verification	Was the system built right against its requirements and design?	Requirements-verification matrix; proofs; inspections; tests; fault injection; red-team results
Transition	Is the verified capability securely established in the fielded environment?	Commissioning and trusted-setup package; as-built reconciliation; authorization evidence
Validation	Is it the right secure capability for real stakeholders and mission use?	End-to-end mission scenarios; measures of effectiveness; stakeholder acceptance; residual risks
Operation	Does active use preserve properties, assumptions, and authorized configuration?	Operational procedures; monitoring and response records; exercises; assurance-case status
Maintenance	Can capability be sustained and changed without creating bypasses?	Maintenance concept; change impact analysis; tool/part provenance; regression evidence; reauthorization
Disposal	Have residual assets, trust relationships, and dependencies reached safe terminal states?	Disposition plan; custody records; sanitization/destruction evidence; credential termination

The controlling governance rule is simple: no security-significant decision or change is allowed to enter at only one point in the system life cycle. A new use case may require renewed stakeholder, requirements, architecture, validation, and operational work. A component replacement may require implementation, integration, verification, transition, and assurance-case updates. A changed adversary assumption may reach all the way back to mission analysis. Tailoring determines the scale of reapplication, but traceability determines the scope. This is how SIDC avoids becoming a strong original design wrapped in progressively weaker system life cycle practices.

6. A Seven-Step Method for Building the Capability

The design principles in SP 800-160 can be converted into a repeatable engineering method. The steps are iterative: discoveries at lower levels can refine losses, hazards, constraints, or assumptions at higher levels.

- **Step 1 — Define the mission capability and claim boundary.** State what the system must do securely, the protected assets and properties, the adversary, the operating period, and the accepted trade-offs. For the SIDC, the mission is sensitive AI inference without unacceptable disclosure or manipulation, even when preserving confidentiality and integrity requires sacrificing availability.
- **Step 2 — Define unacceptable losses, hazards, and constraints.** Start with consequences, not controls. A loss, such as disclosure of model weights, leads to hazards involving the unauthorized transfer, exposure, or reconstruction of model weights and other security-sensitive computational state outside approved trust boundaries. Those hazards, in turn, produce constraints that permit such movement only through explicitly authorized, mediated, and verifiable transactions.[5]
- **Step 3 — Convert constraints into architectural invariants.** Use the principles of *Domain Separation*, *Structured Decomposition and Composition*, *Mediated Access*, and *Trustworthy System Control* to establish rules that must hold in every state. RAND's realm partitioning, fixed diode-gated topology, and verified cross-boundary protocols are exemplars.
- **Step 4 — Optimize each element through composed principles.** Ask whether a function can be eliminated, minimized, isolated, deprived of unnecessary privilege and sharing, mediated, made protectively failing, monitored, restored, and verified. This is an iterative architectural optimization process, not a compliance checklist.
- **Step 5 — Minimize the trusted mechanism.** Apply the principles of *Least Functionality*, *Reduced Complexity*, *Minimal Trusted Elements*, and *Structured Decomposition and Composition* until each security-critical mechanism is sufficiently small, deterministic, and bounded for rigorous analysis. RAND's cross-realm solution demonstrates the progression from constraint to protocol, finite-state behavior, and hardware-realizable logic.
- **Step 6 — Establish the assurance chain.** Trace security objective to loss, hazard, constraint, requirement, architecture, mechanism, implementation, verification, validation, and operational evidence. State assumptions at every link and demonstrate that component evidence composes into system-level claims.
- **Step 7 — Sustain the authorized capability.** Exercise detection, containment, shutdown, restoration, reconstitution, and destructive responses; continuously monitor assumptions, configuration, privileges, anomalies, and evidence validity; and require impact analysis and renewed evidence before changes cross affected trust boundaries.

[5] Computational state can include AI model weights, parameters, and intermediate representations; activations, prompts, inference inputs, outputs, and contextual data; data residing temporarily in processor registers, caches, and memory; snapshots, checkpoints, logs, crash dumps, and diagnostic information; information that could allow an adversary to reconstruct protected model behavior or intellectual property. Movement can occur through file transfers, network traffic, shared memory, storage media, direct memory access, maintenance interfaces, telemetry, covert channels, compromised components, or residual data left after a workload ends.

The seven steps should become decision gates in acquisition, system requirements reviews, preliminary and critical design reviews, verification plans, commissioning, authorization, operations, maintenance, incident exercises, and configuration control. No mechanism or change is accepted because it appears secure; it is accepted because its necessity, constraints, dependencies, failure behavior, and evidence are explicit and tied to prevention of unacceptable loss.

7. Where the NIST SP 800-160 Principles and Processes Extend the RAND Work

RAND is intentionally conceptual and identifies significant implementation backlog: hazard management, recovery and reconstitution, reference-data ingestion, evaluation, and agentic interaction [1]. The NIST SP 800-160 security design principles provide a disciplined way to complete that backlog without diluting the architecture. Three extensions are especially important.

- **First, life-cycle completeness.** RAND demonstrates vertical traceability for selected critical SIDC mechanisms. The NIST SP 800-160 security design principles and system life cycle processes require the same security meaning to persist from mission analysis and stakeholder protection needs through requirements, architecture, design, realization, fielding, use, change, and disposal. The system boundary therefore includes temporary processes, enabling systems, support equipment, people, facilities, and residual assets, not only deployed computing and communications capabilities.
- **Second, explicit composition.** Local assurance is insufficient. Every realm and controller must expose its assumptions and dependencies so the integrated system can demonstrate that no combination of individually acceptable behaviors creates an unacceptable loss. This is essential for cyber-physical flows and human control actions that formal protocol proofs alone cannot cover.
- **Third, evidence governance.** A security claim has an operational lifetime. Evidence must be versioned, protected, reviewed, linked to the fielded configuration, and invalidated when its assumptions change. This governance function is the bridge between RAND’s concept-to-circuit method and a facility that remains trustworthy and resilient throughout its authorized life.

8. Conclusion

RAND's SIDC report makes a persuasive case that exceptionally consequential AI inference workloads may require purpose-built, security-first facilities and that such facilities can be engineered with available technology. Its realm architecture, fixed unidirectional topology, verified cross-boundary protocols, loss-driven requirements, and assurance chain embody many of the strongest ideas in NIST SP 800-160.

The security design principles in NIST SP 800-160 deepen that work by establishing the rules that make the concept operationally credible. The essential contribution is *continuity*: (1) continuity of loss-control intent across abstraction levels; (2) continuity of protection across operating and failure states; (3) continuity of trustworthiness across composition; and (4) continuity of evidence across time and change. *Domain Separation* and *Mediated Access* build the boundaries. *Least Functionality* and *Reduced Complexity* make them analyzable. *Trustworthy System Control* makes them non-bypassable. *Protective Failure* and *Protective Recovery* preserve them under adversity. *Substantiated Trustworthiness* and *Compositional Trustworthiness* make the resulting claims defensible.

The practical test for an SIDC is therefore not simply whether its architecture resembles RAND's diagram or whether selected protocols have been formally verified. The test is whether every function, flow, privilege, transition, exception, recovery action, and change remains traceably necessary, constrained, and supported by evidence commensurate with the losses at stake. Applied in that manner, the NIST SP 800-160 design principles are not supplemental guidance. They are the required *engineering discipline* for turning RAND's SIDC from a compelling design into a trustworthy operational capability.

That engineering discipline must be enacted through the entire system life cycle. *Business or Mission Analysis* justifies the solution and bounds the claim. *Stakeholder Needs and Requirements Definition* and *System Requirements Definition* translate consequences into measurable obligations. *Architecture*, *Design*, *Analysis*, *Implementation*, and *Integration* turn those obligations into a controlled system. *Verification*, *Transition*, and *Validation* establish objective evidence that the right secure capability was built and fielded correctly. *Operation* and *Maintenance* preserve it under use, adversity, and change. *Disposal* ensures that residual assets and dependencies do not outlive their protection. The holistic conclusion is, therefore, stronger than an operational one: an SIDC is trustworthy only when its design principles and its life-cycle processes form one continuous, traceable, evidence-producing system.

Fort Knox is trustworthy not because of a vault door alone, but because its site, structure, access rules, personnel, monitoring, custody, maintenance, and response operate as one protection system. The same standard must govern the SIDC. A “Digital Fort Knox” for critical AI is not merely a hardened facility; it is a continuously engineered, traceable, and evidence-producing system whose trustworthiness is preserved from conception to retirement.

References

[1] S. F. Comer et al., *Secure Inference Data Centers: A Vertically Integrated Strategy for Security Engineering*, RAND Corporation, RR-A4827-1, 2026. https://www.rand.org/pubs/research_reports/RRA4827-1.html

[2] R. Ross, M. Winstead, and M. McEvilley, *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