Mediated Access

Ensuring Every Consequential Interaction Is Authorized and Controlled

Dr. Ron Ross
CEO, RONROSSECURE, LLC
ISTS FELLOW, DARTMOUTH COLLEGE

Abstract

Mediated Access is a foundational design principle for trustworthy secure systems. NIST Special Publication 800-160, Volume 1, Revision 1 states the principle directly: all access to and operations on system elements are mediated. The statement is compact, but its systems engineering implications extend far beyond user logon and conventional access control measures. Mediation determines whether every consequential interaction—human-to-machine, machine-to-machine, component-to-component, cyber-to-physical, administrative, maintenance, recovery, and supply-chain initiated—is subjected to an explicit policy decision and dependable enforcement before the interaction can affect a protected asset or system state. This article examines Mediated Access as a systems security engineering principle. It connects authorization policy to unacceptable loss, interface design, architecture, Engineering for Compromise, complementary design principles, life cycle processes, failure patterns, and assurance evidence. The central proposition is that a system cannot credibly claim control over its behavior when consequential interactions can occur outside its mediation architecture. A trustworthy mediator does more than decide who may enter. It constrains what may be done, to what, under which conditions, for how long, with what effects, and with what evidence.

1. The Engineering Problem

Modern systems depend on *continuous interaction*. People invoke services; components exchange data; software calls APIs; administrators change configurations; sensors influence controllers; automated agents initiate actions; suppliers deliver updates; and recovery services restore system state. These interactions create capability, but they also create the pathways through which error, misuse, failure, and hostile action can produce unacceptable *loss*.

The engineering problem is not simply determining whether an identity is allowed to access a system. It is ensuring that every security-relevant interaction is governed by rules that reflect stakeholder intent and that those rules are enforced at every point by policies and mechanisms where influence can be exercised. A *subject* may be authenticated and still request an unauthorized operation. An authorized administrator may act from an untrusted device. A legitimate service may issue a command at the wrong time, in the wrong system state, or with effects beyond its intended scope. A compromised component may use every privilege it legitimately possesses for malicious purposes.

> ***"Authentication establishes an identity claim. Mediation governs what that identity can do."***

This distinction matters because systems are not harmed by identities in the abstract. Loss occurs through operations and effects: data are disclosed or altered; commands change physical state; resources are exhausted; privileges are delegated; software is installed; recovery images are replaced; safety interlocks are overridden; or a system transitions into a hazardous mode. The *mediation architecture* must therefore control both access to a system element and use of that element.

NIST SP 800-160 defines the primary security objective as ensuring that only intended behaviors and outcomes occur within and with the system. *Mediated Access* is the foundational design principle that

© 2026 RONROSSECURE, LLC | CC BY 4.0.

converts that objective into enforceable decisions. Without mediation, a policy may describe desired behavior, but the architecture cannot dependably prevent *prohibited* behavior.

The challenge becomes more difficult as systems distribute decision-making across autonomous functions, cloud services, operational technology, external providers, machine identities, and human organizations. Multiple *enforcement points* may interpret policy differently. Cached decisions may outlive the conditions that justified them. Emergency and maintenance paths may bypass normal system checks. An apparently comprehensive access-controlled system can therefore contain decisive gaps.

The systems engineering task is to define the *scope of control*, identify every consequential interaction within that scope, express consistent rules for those interactions, allocate decision and enforcement responsibilities, and produce evidence that no material path can bypass or defeat the intended mediation.

2. What the Principle Means

NIST SP 800-160, Volume 1, Revision 1 states the design principle of *Mediated Access* as follows [1]:

> ***"All access to and operations on system elements are mediated."***

Its purpose is to place limits on access to and use of the system, reduce the possibility of loss escalation, and reduce the extent to which loss escalates and propagates [1].

The formulation contains two distinct obligations. First, access to the system element must be authorized: the requesting entity may reach only the target elements it is permitted to reach. Second, use of the system element must be authorized: the requesting entity may perform only the operations it is permitted to perform. Allowing a subject to reach an object does not imply permission to exercise every operation the object supports.

Mediated Access contains two engineering functions: a policy-based *decision*, and *enforcement* of that decision. The access control decision evaluates the request against applicable rules and current conditions. Access enforcement ensures that the interaction occurs only as decided. Either engineering function can fail independently. A correct decision without non-bypassable enforcement is ineffective; strong enforcement of inaccurate, inconsistent, or incomplete policy can reliably produce the wrong result.

"Mediation is trustworthy only when the decision is correct and the enforcement is unavoidable."

The rules may consider subject, object, requested operation, role, attributes, provenance, time, location, system state or mode, device condition, risk condition, mission phase, separation of duties, duration, rate, sequence, and prior actions. Mediation is therefore broader than a static allow-or-deny list. It can govern the vocabulary, direction, timing, capacity, and state transitions permitted through a system interface.

The design principle of *Mediated Access* applies to human and machine entities alike. Subjects include users, administrators, services, applications, devices, autonomous agents, processes, maintenance tools, suppliers, and external systems. Objects include data, functions, components, actuators, models, keys, configurations, physical resources, logs, recovery artifacts, interfaces, and other subjects. Subject-to-subject interactions can be as consequential as subject-to-object access when one entity can direct, impersonate, delegate to, or reconfigure another.

A complete mediation architecture establishes a defined *scope of control*. Within that scope, every security-relevant path is known; policies are accurate, consistent, compatible, and sufficiently complete; decision

© 2026 RONROSSECURE, LLC | CC BY 4.0.

and enforcement mechanisms are protected; and the system maintains *Mediated Access* across normal, degraded, maintenance, emergency, recovery, and transition states.

3. What the Principle Does Not Mean

Mediated Access is frequently narrowed to familiar identity and access-management functions. That interpretation is incomplete.

- **Mediated Access does not mean authentication alone.** Authentication supports a claim about identity or origin. It does not determine whether the requested target, operation, timing, context, or effect is authorized.
- **Mediated Access does not mean one authorization check at session establishment.** Authorization conditions can change during a session. Complete mediation may require reevaluation for each consequential operation or whenever relevant state changes.
- **Mediated Access does not mean role-based access control alone.** Roles may be useful, but role membership can be too coarse. Attributes, relationships, system state, mission phase, risk, provenance, and operation-specific constraints may also be necessary.
- **Mediated Access does not mean network admission.** A firewall, gateway, or zero-trust access broker can mediate a connection while application operations, control commands, data uses, resource consumption, or physical effects remain insufficiently constrained.
- **Mediated Access does not mean that trusted insiders, administrators, or services are exempt.** High privilege increases the consequence of compromise and misuse; it strengthens rather than removes the need for mediation.
- **Mediated Access does not mean that policy enforcement can be bypassed for performance or availability without changing the security claim.** Cached decisions, direct database access, service accounts, emergency interfaces, and maintenance channels are part of the mediation architecture.
- **Mediated Access does not mean deny everything.** The system must reliably permit authorized mission behavior as well as prevent prohibited behavior. A mediator that blocks required capability is not an adequate engineering solution.
- **Mediated Access does not mean a security product is trustworthy merely because it is marketed as an access-control solution.** Its correctness, tamper resistance, non-bypassability, failure behavior, dependencies, configuration, and operational administration must be substantiated.

The principle is best understood as a system property: throughout the relevant scope and life cycle states, every consequential interaction is subjected to policy-based control before it can produce an effect.

4. Loss-Driven Interpretation

A *mediated-access architecture* should begin with unacceptable loss, not with identities, products, or policy statements. Stakeholders depend on system capabilities and assets; those dependencies create adverse consequences that cannot be tolerated. The engineering question is which interactions could create or enable the system conditions that lead to those consequences.

A loss-driven chain for *Mediated Access* can be expressed as:

Asset → Unacceptable Loss → Hazardous or Loss-Enabling Interaction → Security Constraint → Policy Rule → Decision and Enforcement → Evidence.

© 2026 RONROSSECURE, LLC | CC BY 4.0.

Consider the water treatment facility used across this series, with two process lines, local control, supervisory applications, and a common reporting service. The unacceptable outcomes are release of unsafe treated water and loss of minimum required treatment service across both lines from a disturbance intended to be contained to one. A supervisory application may request dosing changes, but each consequential request must be evaluated for its target, operation, parameters, sequence, and relevant process state before enforcement permits the effect. Reporting authority does not confer control-write authority.

The command control mechanism comprises the functions that evaluate a supervisory request and make the resulting restriction effective. In NIST terms, distinguish protection control decision, enforcement, and infrastructure functions [1]. Rejection or loss of a supervisory service must preserve specified local control or a validated protective mode. Facility-specific analysis must establish the permitted operating envelope, decision inputs, response times, and minimum-service behavior; indiscriminate rejection can itself create loss.

Now consider a critical AI inference environment. Unauthorized extraction of model weights may be an unacceptable loss. A request may come from a legitimate inference service, but that service should be authorized only to invoke a narrow inference operation, not to read model memory, export weights, alter execution policy, retrieve keys, or access recovery artifacts. Mediation translates the distinction between intended use and prohibited influence into *enforceable interface behavior*.

"The question is not only who may connect. It is what effects the system will permit that connection to produce."

Loss-driven reasoning also determines mediation *strength*. A low-consequence resource may tolerate a centralized general-purpose authorization service. A catastrophic control action may require independent enforcement, distributed approval, hardware protection, restrictive defaults, and assurance evidence commensurate with the *consequence of failure*. The design principle of *Commensurate Protection* prevents uniform controls from being applied where the stakes differ materially.

This framing exposes incomplete policies. If the unacceptable loss can occur through timing, resource exhaustion, command sequence, delegation, update, maintenance, recovery, or physical access, a policy that governs only read, write, and execute operations is insufficient. The scope of mediation must follow the causal paths to *loss*.

5. Engineering the Principle into the Architecture

Architects realize *Mediated Access* through a coordinated set of decisions. The following actions guide and inform the engineering process:

- **Define the scope of control.** Identify the assets, system elements, environments, people, external systems, and life cycle states governed by each policy. Ambiguous scope creates gaps between enforcement domains.
- **Enumerate subjects, objects, and consequential operations.** Include machine identities, services, autonomous agents, administrators, maintenance tools, recovery functions, suppliers, APIs, control commands, configuration changes, physical actions, and delegation.
- **Derive policy from stakeholder protection needs and system security requirements.** Policy must reflect what behavior and outcomes are authorized, not merely reproduce existing permissions or organizational habits.

© 2026 RONROSSECURE, LLC | CC BY 4.0.

- **Separate policy decision from policy enforcement conceptually, even when implemented together.** State what information the decision requires, how it is obtained, where enforcement occurs, and what happens if either function is unavailable or inconsistent.
- **Place enforcement where effects can actually be controlled.** A decision made at the network edge may not constrain a privileged application operation or physical actuator. Mediation points should dominate the paths by which prohibited effects can occur.
- **Make the mediation path non-bypassable.** Direct interfaces, alternate protocols, cached credentials, local consoles, diagnostic ports, batch processes, replication channels, emergency accounts, and recovery tools must be included or explicitly constrained.
- **Constrain more than identity and object.** Where relevant, control operation, parameters, direction, sequence, frequency, capacity, duration, state transition, delegation, and downstream use.
- **Apply restrictive defaults.** Missing, conflicting, outdated, or unverifiable policy information should move the system toward a protective state rather than silently authorize the request.
- **Protect the mediator and its dependencies.** Policy stores, identity evidence, time sources, update mechanisms, cryptographic keys, telemetry, administrative interfaces, and decision inputs become part of the trusted footprint.
- **Design for policy change.** Versioning, distribution, activation, rollback, synchronization, conflict resolution, and revocation must preserve consistent enforcement across distributed system elements.
- **Engineer performance deliberately.** Because mediation can become a bottleneck, capacity and latency must be treated as requirements. Optimization must not create unexamined bypasses or out-of-date authorization.
- **Make decisions observable and explainable.** Logs should capture the request, relevant decision context, applicable policy, result, enforcement point, and material failures without disclosing protected information.

The architectural test is straightforward: for every path capable of producing a security-relevant effect, identify the governing policy decision and the mechanism that makes the decision unavoidable.

6. Engineering for Compromise

Engineering for Compromise strengthens the mediated-access question [2]. Conventional access-control design often assumes that authenticated subjects and authorized components will use granted privileges benignly. *Engineering for Compromise* assumes a selected subject, system component, credential, decision service, or enforcement point may be hostile and asks what *authority* the adversary inherits.

A compromised system element can exercise every operation legitimately available to it. Mediation therefore determines the adversary's post-compromise action space. If an externally exposed service can invoke a general administrative API, access plaintext keys, or submit unconstrained control parameters, compromise of that service can make unacceptable loss immediately reachable. If it can invoke only a narrow, state-checked operation through an independently protected mediator, compromise yields a foothold but not unrestricted authority.

In the water treatment facility, assume the supervisory application is compromised. It retains every operation it is authorized to request. The mediation question is whether those operations, issued in any order, with any permitted parameters, and at any time, can drive either process line outside its validated operating envelope or interrupt minimum required treatment service across both process lines. Mediation

© 2026 RONROSSECURE, LLC | CC BY 4.0.

at local control, evaluated against current process state, bounds what the compromised application can achieve.

> ***"The privileges granted to a system element today become the adversary's capabilities if that system element is compromised tomorrow."***

For each subject or system element, engineers should ask: assume this entity is completely controlled by the adversary. Which objects can it reach? Which operations can it invoke? Which conditions can it manipulate? Can it delegate authority, suppress evidence, alter policy, exhaust the mediator, or reach an alternate path? Which combinations of allowed operations produce a prohibited outcome even when each operation appears individually legitimate?

This analysis reveals the difference between policy correctness and policy adequacy. A mediator may enforce its configured rules perfectly while those rules grant a compromised subject enough authority to cause catastrophic loss. *Engineering for Compromise* therefore feeds compromise consequences back into privilege allocation, interface design, operation granularity, distributed approval, and domain structure.

Mediated Access also contributes to *Compromise Containment Boundaries*. The design principle of *Domain Separation* establishes where influence should change; mediation controls the necessary crossings. The principle of *Least Privilege* bounds the operations available after crossing. The principle of *Distributed Privilege* can make one compromised subject insufficient for catastrophic action. The principle of *Least Sharing* reduces the common services through which authorization or compromise can propagate.

The mediator itself must also be included in the compromise analysis. If compromise of one policy administrator, one decision service, or one signing key permits universal authorization, the architecture has concentrated catastrophic capability. Independent access enforcement, protected policy roots, dual authorization, threshold mechanisms, diversity, and recovery from known-good policy may be warranted for high-consequence decisions.

Compromise need not reach the mediator to defeat it. A mediator decides from the identity, attributes, process state, time, and context supplied to it. If a compromised system element can supply or alter those inputs, it can obtain an authorization the policy was written to prevent. In the water treatment facility, a compromised supervisory application that can report false process conditions may make an unsafe dosing request appear consistent with the current state of the process line. Inputs to high-consequence decisions should come from sources the requesting subject cannot control, and the argument should state which inputs remain trustworthy under the stated compromise assumptions.

The decision must also govern the operation that actually takes effect. A compromised subject may obtain approval for one request and execute another, alter parameters after the check, replay an earlier authorization, or route its request through a more privileged service that acts on its behalf. Mediation binds each decision to the specific subject, operation, target, parameters, and time window; enforces it at or near the point of effect; and evaluates a delegated request against the authority of the originating subject, not only that of the intermediary.

Engineering for Compromise also examines *failure behavior*. A fail-open mediator preserves availability by abandoning the security constraint. A fail-closed mediator may create mission or safety loss by denying essential action. The design principle of *Protective Failure* requires a context-sensitive design: the system may need bounded local authority, a safe operational mode, preauthorized emergency actions, or independent access controls that preserve both mission capability and the most critical constraints.

© 2026 RONROSSECURE, LLC | CC BY 4.0.

Compromise Propagation Analysis should follow mediation dependencies as carefully as application connections. Shared identity, common policy services, synchronized credentials, administrative planes, and authorization libraries can cause one compromise to affect many established domains. A meaningful mediation architecture changes adversary reachability; it does not merely add another software call along the same trust path.

Compromise also changes what revocation and recovery must achieve. Once a subject is suspected, its authority must be withdrawn at every enforcement point within the time needed to prevent loss, including cached decisions and sessions already established. If the mediator, its policy, or its keys may be compromised, recovery restores them from a trusted representation and reestablishes the authorization state deliberately rather than resuming from a state the adversary may have shaped.

These considerations define the compromise assumptions of the mediation claim. The argument should state which subjects, services, and dependencies may be compromised without invalidating the claim, which must remain trustworthy, and which independent constraints, such as the local protective functions on each process line, continue to prevent unacceptable loss if mediation itself fails. Verification then exercises those assumptions directly: hostile-subject testing of every granted operation, manipulation of decision inputs, replay and substitution of approvals, and attempts to reach the protected resource without invoking the mediator.

7. Interaction with Other Principles

The principle of *Mediated Access* is foundational, but it cannot stand alone. Its strength depends on composition with other NIST security design principles.

- **Domain Separation.** Establish the boundaries at which interaction must be controlled. Separation without mediation prevents useful cooperation or invites bypass; mediation without meaningful separation may leave enforcement embedded in the same compromise domain as the subject it constrains.
- **Least Privilege.** Determine the maximum authority a successful decision should confer. Mediation enforces the allocation; *Least Privilege* limits the damage available through correct or stolen authorization.
- **Least Sharing.** Reduce common policy, identity, administrative, and resource dependencies. Unnecessary sharing expands the blast radius when a mediator or its supporting services are compromised. SP 800-160 treats least common mechanism as an instance of *Least Sharing* rather than as a separate principle, so common mediation dependencies are analyzed here.
- **Distributed Privilege.** Require multiple independent parties or system elements for selected consequential actions. *Mediated Access* coordinates and enforces the combined authorization so that no single compromise is sufficient.
- **Protective Defaults.** Ensure that absence of explicit authorization, missing context, policy conflict, or uncertainty results in a protective decision.
- **Least Functionality.** Narrow the operations that the mediator must understand and govern. An interface that exposes unnecessary functions creates unnecessary authorization complexity and adversary capability.
- **Least Persistence.** Limit when privileges, sessions, credentials, services, and paths remain available. Mediation can constrain authorization by time, state, duration, and continuing need.

© 2026 RONROSSECURE, LLC | CC BY 4.0.

- **Minimal Trusted Elements.** Reduce the set of mechanisms whose failure can invalidate the mediation claim. Decision and enforcement functions should be small enough to analyze and protect commensurately.
- **Trustworthy System Control.** Apply the generalized reference monitor criteria to mediation mechanisms and protect their policy, decision inputs, invocation, and enforcement paths.
- **Reduced Complexity.** Make policy and enforcement behavior analyzable. Excessively expressive policy, overlapping rules, recursive delegation, and opaque inheritance can conceal contradictions and gaps.
- **Continuous Protection.** Require mediation to persist during startup, degraded operation, maintenance, update, recovery, and shutdown, not only during routine operation.
- **Protective Failure and Protective Recovery.** Govern the mediator under fault or attack and restore an authenticated policy, trustworthy decision basis, and correct enforcement state.
- **Anomaly Detection.** Use observations of consequential operations, policy state, and enforced results to identify salient departures. Denied requests are one source of evidence; an allowed request can still have an unacceptable effect. Protect the observations, account for missing or outdated data, and provide enough time and authority for the required response.
- **Substantiated Trustworthiness and Compositional Trustworthiness.** Require evidence that individual decision and enforcement system elements are trustworthy and that their composition preserves complete, correct, non-bypassable mediation.

The broader lesson is that *Mediated Access* is the enforcement center of an architectural system of principles. The supporting principles determine what must be mediated, how much authority may be granted, how decisions fail and recover, and what confidence can be placed in the result.

8. System Life Cycle Application

Mediated Access must be preserved throughout the system life cycle, not added during deployment. NIST SP 800-160, Volume 1, Revision 1 organizes thirty system life cycle processes into four groups: fourteen *Technical Processes*, eight *Technical Management Processes*, six *Organizational Project-Enabling Processes*, and two *Agreement Processes* [1]. The fourteen Technical Processes provide the focus below.

- **Business or Mission Analysis.** This process identifies mission dependencies, unacceptable losses, authorities, operational contexts, and consequential interactions that require control.
- **Stakeholder Needs and Requirements Definition.** This process captures who or what may influence critical assets, which outcomes are prohibited, and what availability, safety, privacy, and accountability constraints apply.
- **System Requirements Definition.** This process converts those needs into verifiable requirements for access, context, operations, decision latency, enforcement coverage, failure, recovery, audit, and assurance.
- **System Architecture Definition.** This process allocates subjects, objects, domains, policy authorities, decision functions, enforcement points, identity services, trusted system elements, and interfaces.
- **Design Definition.** This process specifies policy models, protocols, operation granularity, decision inputs, privilege structures, cryptographic protections, administrative controls, synchronization, and failure behavior.
- **System Analysis.** This process evaluates policy completeness, rule conflicts, bypass paths, trust dependencies, performance, availability, common-mode failure, compromise propagation, and alternative architectures.

© 2026 RONROSSECURE, LLC | CC BY 4.0.

- **Implementation.** This process realizes mediation correctly in hardware, software, firmware, physical arrangements, configurations, and procedures. Secure coding alone is insufficient if enforcement placement is incomplete.
- **Integration.** This process verifies that independently designed services interpret identity, attributes, policy, operations, and error states consistently. Integration often creates direct paths and privileged service relationships absent from component designs.
- **Verification.** This process demonstrates that authorized operations succeed, prohibited operations fail, conditions are evaluated correctly, bypass paths are absent within the claim scope, and faults produce specified behavior.
- **Transition.** This process controls provisioning, policy activation, credential issuance, data migration, initial trust establishment, and temporary deployment access.
- **Validation.** This process determines whether the mediation architecture supports stakeholder protection needs and mission performance in realistic operations, including degraded and emergency conditions.
- **Operation.** This process monitors decisions, denials, privilege use, configuration drift, policy changes, capacity, attacks on decision services, and evidence that assumptions remain valid.
- **Maintenance.** This process governs vendor access, diagnostics, patching, testing, emergency authority, component replacement, and policy updates. Temporary exceptions must be explicitly authorized, bounded, observed, and removed.
- **Disposal.** This process revokes identities and privileges, removes policy and key material, sanitizes decision records, terminates trust relationships, and prevents decommissioned system elements from retaining access.

The governing rule is *continuity of authorization intent*: stakeholder decisions about permissible behavior must survive translation into requirements, architecture, implementation, operation, change, recovery, and retirement.

"The governing rule is continuity of authorization intent."

---

9. Failure Patterns and Anti-Patterns

Systems frequently appear to mediate access while consequential interactions remain uncontrolled. Several failure patterns recur.

- **The authentication-equals-authorization assumption.** A valid identity or device certificate is treated as sufficient permission for broad operations. The system proves who is acting but fails to constrain what the actor may do.
- **The session-front-door check.** Authorization occurs when a session is established, but later operations, privilege changes, context changes, and long-lived connections are not reevaluated.
- **The incomplete mediation map.** Primary APIs are governed, but batch jobs, local consoles, database connections, message queues, replication, diagnostic ports, recovery interfaces, and physical controls remain alternate paths.
- **The overbroad service identity.** Many workloads share one powerful device identity. Compromise of any workload inherits the full authority of the shared account and obscures attribution.

© 2026 RONROSSECURE, LLC | CC BY 4.0.

- **The policy-enforcement mismatch.** Central policy describes fine-grained constraints, but downstream systems can enforce only coarse roles or network location. The apparent policy exceeds actual protection capability.
- **The out-of-date-decision problem.** Cached authorizations, long-lived tokens, disconnected operation, or delayed revocation allow access after role, risk, device, mission, or employment conditions have changed.
- **The policy-administration bypass.** The operational interface is strongly mediated, but one administrator, automation pipeline, or compromised update process can rewrite policy or disable enforcement.
- **The fail-open availability shortcut.** When identity, policy, time, telemetry, or decision services are unavailable, the system permits requests to preserve service without evaluating the consequence of abandoned constraints.
- **The emergency-access permanence problem.** Break-glass access is broad, weakly monitored, persistently enabled, or never revoked after use. The exceptional mechanism becomes the preferred adversary path.
- **The individually-authorized sequence.** Each operation is allowed in isolation, but a sequence of permitted actions creates a prohibited state or bypasses an intended limit.
- **The policy-conflict blind spot.** Multiple policy authorities issue rules that overlap, contradict, or leave gaps. Enforcement order or local interpretation determines outcomes that stakeholders never explicitly authorized.
- **The trusted-mediator fallacy.** The organization assumes that a commercial gateway or authorization engine is inherently trustworthy and does not analyze its dependencies, configuration, administrators, update path, or compromise consequences.

These failure patterns share a common feature: the visible control point is not the complete boundary of authorized influence. Engineering analysis must follow every path by which an interaction can produce a consequential effect.

> ***"A request is not fully mediated when the same effect remains reachable by another path."***

10. Evidence and Assurance

A *Mediated Access* claim is meaningful only when supported by *evidence* commensurate with the consequences of mediation failure and asset loss. The existence of a policy engine, gateway, identity provider, or authorization library is not sufficient. Evidence must support the system-level conclusion that all relevant interactions are governed by accurate policy and subject to dependable enforcement.

The claim and subclaims below are an illustrative assurance pattern. They state propositions to substantiate for a particular system, not findings that the system already satisfies them. Evidence types identify what to obtain and evaluate; they are not completed assurance evidence. Within a wider system assurance case, this principle-specific claim can support a higher-level claim about unacceptable loss.

A top-level claim might be:

© 2026 RONROSSECURE, LLC | CC BY 4.0.

Within the defined system scope, access to and operations on system elements are subject to the specified policy decisions and non-bypassable enforcement under the stated operating conditions, life cycle transitions, and failure and compromise assumptions.

The following supporting subclaims organize the assurance argument and identify the engineering evidence needed to support it.

- **Subclaim 1: Scope and policy obligations are complete within the defined scope.** Subjects, objects, consequential operations, equivalent effects, and relevant states are identified, including administration, maintenance, recovery, and automated agents. Evidence includes requirements traceability, loss and hazard analyses, subject–object–operation inventories, and architecture and flow models.
- **Subclaim 2: Decision rules and inputs are adequate for the specified policy obligations.** Rules are consistent and sufficiently complete for the constraints in scope; identity, attributes, parameters, process state, sequence, timing, and context are valid for the request and consequential decision. Evidence includes policy specifications, consistency and coverage analyses, identity-binding and input-validity evaluations, privilege matrices, and formal analysis where justified.
- **Subclaim 3: Enforcement governs every relevant path within scope.** Direct, indirect, administrative, maintenance, and recovery paths receive the required restriction at the point of effect. Composed decision and enforcement functions preserve non-bypassability, evaluatability, required invocation, and protection against unauthorized modification. Evidence includes reference-monitor design arguments, interface and implementation analysis, composition analysis, and positive, negative, and bypass tests.
- **Subclaim 4: The mediator and its dependencies are protected under the stated assumptions.** Decision, enforcement, infrastructure, policy administration, identity, keys, updates, and supporting data remain dependable to the extent required by the claim. The argument identifies trusted dependencies and the restrictions that survive each tolerated compromise. Evidence includes implementation review, configuration baselines, administrative privilege analysis, and compromise-oriented tests.
- **Subclaim 5: Required behavior and Protective Failure satisfy the specified constraints.** Authorized mission operations remain achievable, malicious use of granted operations is constrained as specified, revocation takes effect within its required delay, and faults produce the defined protective and minimum-service behavior. Evidence includes functional tests, performance and capacity analysis, hostile-subject exercises, revocation measurements, and fault injection.
- **Subclaim 6: Mediation is preserved in operation, through change, and in recovery.** Operational use, updates, emergency access, new subjects or operations, and restored state do not introduce unexamined exceptions or bypasses. Evidence includes operational records, configuration and change analysis, policy activation and rollback tests, maintenance and emergency-mode exercises, restoration demonstrations, and assurance-case updates.

The argument connects complete policy scope, adequate decisions, and unavoidable protected enforcement to governed interactions. Prevention of a specified mission loss also requires evidence that the policy and remaining physical and logical constraints address its loss pathways. Authentication, authorization, and correct mediation alone do not establish safe treatment.

Verification should include positive and negative cases. It must show that authorized mission operations succeed and that prohibited operations, malformed requests, unauthorized sequences, out-of-date credentials, missing attributes, conflicting rules, and bypass attempts are rejected or moved into a defined protective state. High-consequence claims may require evidence about absence: not merely that the intended interface works, but that no material alternate path exists within the declared scope.

© 2026 RONROSSECURE, LLC | CC BY 4.0.

Trustworthy System Control provides a design assurance basis for the mediation mechanism [1]. Apply the four generalized reference monitor criteria: non-bypassable, evaluatable, always invoked when the constraint applies, and tamper-proof. NIST SP 800-160 distinguishes the historical reference validation mechanism criteria concerning tamper protection, invocation, and rigorous analysis and tests from the explicitly added non-bypassability criterion. The historical reference monitor work provides additional context [3]. In a distributed realization, evidence must establish that the composed decision, enforcement, and infrastructure functions preserve the required criteria across the relevant services, domains, and states.

In the water treatment example, assume one supervisory application is hostile while the identified command control mechanism, necessary process data sources, local control, and independent protective functions remain trusted. Exercise valid but harmful parameter combinations, repeated requests, alternate maintenance paths, and unavailable decision inputs. Observe the command accepted at local control and its effect on the process. Demonstrate that rejection and overload preserve the specified service and protection. A log of a denied request supports only part of that argument.

An *assurance case* makes the reasoning explicit. It records the claim, decomposes it, links evidence to the supporting argument, and identifies assumptions and external dependencies. It should state which system elements must remain trustworthy, which combinations of compromise exceed the design basis, and which operating modes reduce or alter mediation.

"The presence of an enforcement point is evidence of implementation—not proof of complete mediation."

Evidence must be maintained as the system evolves. New APIs, AI agents, cloud services, roles, suppliers, operational modes, maintenance procedures, or recovery paths can create new subjects, objects, bypass opportunities, and operations. Policy and system architecture changes should trigger renewed analysis, verification, and assurance-case updates.

For *Engineering for Compromise*, assurance should include hostile-subject testing: assume an authorized system element is malicious and exercise every granted operation, parameter range, sequence, delegation path, and timing opportunity. The question is whether the mediation architecture bounds the resulting authority below the threshold of *unacceptable loss*.

11. Key Engineering Takeaways

The following takeaways summarize the engineering decisions and assurance obligations needed to apply the design principle of *Mediated Access*.

1. *Mediated Access* governs both access to system elements and the operations performed on them; authentication alone is insufficient.
2. Every consequential human, machine, software, administrative, maintenance, recovery, and physical interaction within the defined scope must be subject to policy-based decision and dependable enforcement.
3. Mediation begins with unacceptable loss. Policy should constrain the interactions and effects that create loss-enabling system conditions.
4. A correct decision is ineffective without non-bypassable enforcement, and strong enforcement is dangerous when policy is inaccurate, inconsistent, incomplete, or misaligned with stakeholder intent.

© 2026 RONROSSECURE, LLC | CC BY 4.0.

5. Mediation can include identity, operation, parameters, direction, sequence, rate, capacity, timing, duration, system state, mission phase, risk, provenance, and delegation.
6. *Engineering for Compromise* requires designers to treat every privilege granted to a component as capability potentially inherited by an adversary.
7. The design principles of *Domain Separation*, *Trustworthy System Control*, *Least Privilege*, *Least Sharing*, *Distributed Privilege*, *Protective Defaults*, *Minimal Trusted Elements*, *Reduced Complexity*, *Continuous Protection*, *Protective Failure*, *Protective Recovery*, *Compositional Trustworthiness*, and *Substantiated Trustworthiness* strengthen the mediation architecture.
8. The mediator, policy authority, identity evidence, administrative plane, keys, time sources, decision inputs, update mechanisms, and recovery services are part of the trusted footprint.
9. Mediation must persist across normal, degraded, emergency, maintenance, update, recovery, transition, and disposal states.
10. The strongest assurance question is not whether an access-control mechanism exists. It is whether every path capable of producing a prohibited effect is governed by an enforceable, trustworthy policy decision.

The principle of *Mediated Access* provides the control function that makes authorization real.

12. Conclusion

Mediated Access is a deceptively concise principle with system-wide consequences. NIST SP 800-160 states that all access to and operations on system elements are mediated. Realizing that principle requires much more than authenticating users or deploying an access-control product. It requires a coherent architecture of policy, decisions, enforcement, trust, failure behavior, recovery, and evidence.

The principle connects stakeholder intent to *system behavior*. Unacceptable losses define the effects that must be prevented. Security constraints establish what must occur and must not occur. Policy expresses authorized interactions. Decision functions evaluate requests in context. Enforcement mechanisms ensure that only approved behavior can produce effects. Assurance provides justified *confidence* that this chain remains intact.

Engineering for Compromise makes *Mediated Access* indispensable. Once a subject or system component is assumed hostile, every privilege and interface available to it becomes adversary capability. Mediation limits that capability by constraining what the compromised system element can reach, invoke, change, delegate, and cause. It can turn a successful penetration into a bounded event rather than an immediate path to mission loss.

The mediator must be treated as a high-consequence system element. If it is bypassable, overly complex, broadly shared, poorly administered, or dependent on untrustworthy inputs, the authorization claim collapses. If its policy is incomplete, the system may reliably enforce the wrong boundary. Trustworthiness therefore depends on both the *correctness* of the rule and the *integrity* of the mechanism that makes the rule unavoidable.

A trustworthy secure system is not one that merely knows who is present. It is one that controls what every entity can do, under the conditions that matter, for as long as the authority is justified, and with evidence sufficient to support confidence in the result.

© 2026 RONROSSECURE, LLC | CC BY 4.0.

"The engineering purpose of mediated access is to ensure that no consequential interaction escapes authorized control."

Acknowledgments

The author gratefully acknowledges the support provided by Dartmouth College's Institute for Security, Technology, and Society (ISTS). The resources provided by the ISTS contributed to the research and development of this paper. The views and conclusions expressed herein are those of the author and do not necessarily represent the views of Dartmouth College.

Revision 1 (November 2026). Revised for consistency across the series: terminology, the recurring water treatment example, reciprocal relationships among the design principles, and the Section 10 assurance claim and six supporting subclaims. The reference monitor criteria are now attributed to their sources. Section 6 was expanded to address manipulation of decision inputs, binding of decisions to the operations that take effect, and revocation and recovery under compromise.

References

[1] 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

[2] R. Ross, "Engineering for Compromise: Designing Systems That Remain Trustworthy Secure Under Adversity," RONROSSECURE, LLC, August 2026. https://ronrossecure.com/library

[3] J. P. Anderson, Computer Security Technology Planning Study, Volume II, Electronic Systems Division, Air Force Systems Command, 1972.

© 2026 RONROSSECURE, LLC | CC BY 4.0.