Scientific Research Publishing

Journal of Information Security, 2026, 17(3), 322-370
https://www.scirp.org/journal/jis
ISSN Online: 2153-1242
ISSN Print: 2153-1234

Optimization of Zero Trust Principles in Modern Security Operations: A Framework of Minimum Standing Trust

Peter J. Worth Jr.[1,2], Chase Cunningham[1,3], Ionut Cardei[1,2]

[1]Athena Security Group, West Palm Beach, FL, USA
[2]Department of Electrical Engineering and Computer Science, Florida Atlantic University, Boca Raton, FL, USA
[3]Absolute Zero Cyber LLC, Nokesville, VA, USA
Email: pworthjr@athenasecuritygrp.com

How to cite this paper: Worth Jr., P.J., Cunningham, C. and Cardei, I. (2026) Optimization of Zero Trust Principles in Modern Security Operations: A Framework of Minimum Standing Trust. *Journal of Information Security*, **17**, 322-370.
https://doi.org/10.4236/jis.2026.173016

Received: May 30, 2026
Accepted: July 12, 2026
Published: July 15, 2026

Copyright © 2026 by author(s) and Scientific Research Publishing Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/

Open Access

Abstract

Regulated SaaS providers must defend exposed web applications and APIs, distributed cloud control paths, BYOD-heavy workforces, and privileged internal workflows while preserving availability, auditability, and user experience. Zero Trust Architecture (ZTA) is widely adopted for these environments, yet practical implementations frequently devolve into either an unworkable internal deny-all posture or an insufficiently rigorous configuration that leaves unjustified standing trust intact. This paper proposes the Minimum Standing Trust (MST) framework: an operations optimization model for SOCs supporting SaaS platforms that process regulated data (PHI, PII, PCI-DSS-scoped, GDPR-governed). MST formalizes a two-plane enforcement architecture: at the application edge, strict minimization of allowed ingress through WAF controls, NIDS visibility, bidirectional abuse-list auto-blocking at both network and endpoint layers, and governed exception handling; inside the distributed enterprise, the progressive removal of standing trust through strong identity, device posture, session-aware authorization, just-in-time privilege elevation, and behavioral context. The operational hinge is a unified telemetry plane spanning SIEM, EDR, XDR, WAF, NIDS, IAM, VPN/ZTNA, cloud, and data-access events, on which agentic AI can safely accelerate triage and response—but only with autonomy bounded by policy, reversibility, and blast-radius constraints derived from organizational risk tolerance. The framework's anticipated effects—reduced analyst burden, less unnecessary standing access, and improved audit-evidence generation—are stated as explicit, measurable hypotheses tied to defined metrics and baselines rather than as demonstrated results, with empirical validation identified as the priority for future work. The result is a concrete path for aligning Zero Trust theory with cyber defense practice in regulated SaaS environments.

Keywords

Minimum Standing Trust, Zero Trust Architecture, Security Operations Optimization, Telemetry Fusion, Identity-Centric Access, Agentic AI Governance, Reversible Response, Regulated SaaS, Virtual Security Operations Center

1. Introduction

NIST Special Publication 800-207 defines Zero Trust as a shift away from static, network-based perimeters toward controls centered on users, assets, and resources, framed explicitly as a response to remote users, BYOD programs, and cloud-hosted assets [1]. That description maps precisely onto the modern regulated SaaS provider—and generalizes to any organization operating an externally facing application infrastructure alongside an internally facing enterprise network: the two fundamental planes in our model. In such environments, the application infrastructure lives in the cloud; identities and sessions traverse public and private networks; support, engineering, and operations are distributed across geographies and devices; and a constrained but consequential subset of internal roles requires governed access to regulated records. The protected resource is not a building or a subnet; it is the intersection of application paths, identities, sessions, data stores, and administrative workflows that process or expose PHI, PII, PCI-DSS-scoped, GDPR-scoped, and similarly high-consequence information.

Figure 1 illustrates the environment that motivates the MST framework. The key structural insight is that a regulated SaaS provider operates two very different exposure surfaces simultaneously—an Internet-facing application plane serving customers and partners, and an internal enterprise plane of distributed employees, contractors, and administrative workflows—and that the enforcement logic appropriate for one is wrong for the other. This two-plane asymmetry recurs throughout the paper.

The Zero Trust principle traces its formal expression to the foundational Forrester Research report [2], which argued that the "trusted inside" assumption was already empirically false for enterprises with mobile workers, contractors, and cloud-resident data. Google's BeyondCorp operationalized the intuition at scale, moving access controls from the network perimeter to the individual device and user [3] [4], and Gilman and Barth codified the design principles for a wider audience [5]. NIST publications [1] [6] elevated these ideas into normative federal guidance, CISA's Zero Trust Maturity Model [7] provided a staged roadmap, and the DoD Zero Trust Strategy [8] and Executive Order 14028 [9] institutionalized adoption as national security policy.

Despite this rich lineage, many Zero Trust programs become operationally brittle because they are treated as procurement exercises—"Zero Trust products" layered onto unchanged operating models—rather than as a change in how trust de-

cisions are made, evidenced, and governed. The result is coverage without posture: controls exist, but standing privilege persists beneath them, telemetry stays fragmented, and exceptions accumulate without review. The gap this paper addresses is operational: running Zero Trust as a continuous optimization discipline rather than a deployment milestone.

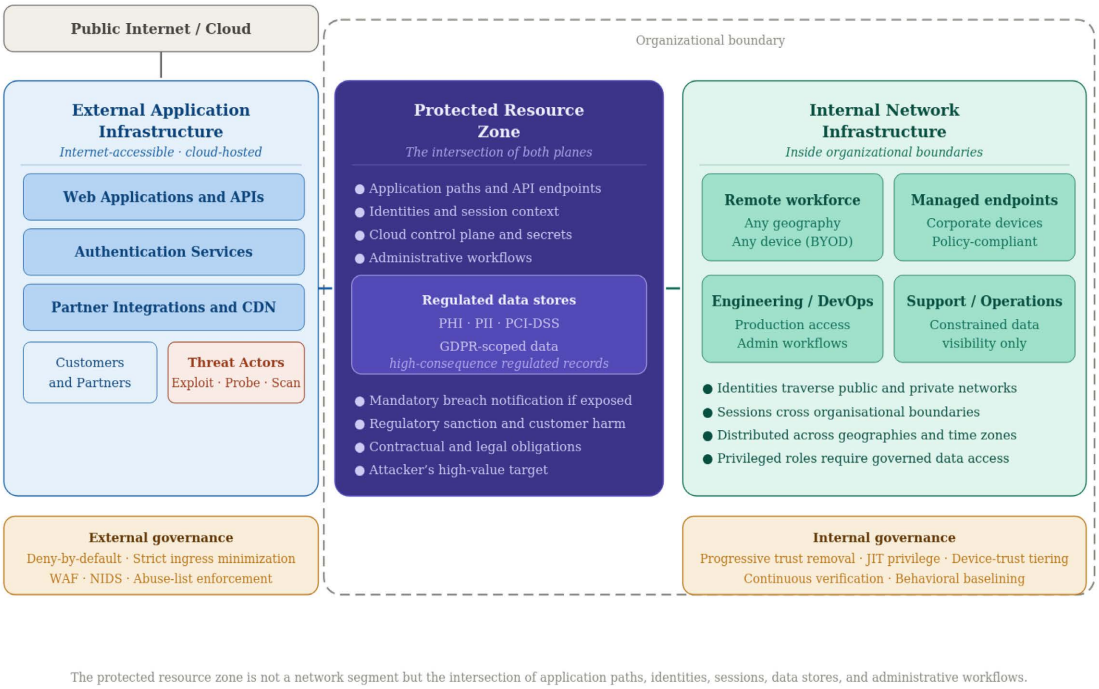


Figure 1. The two-plane operating environment. The external application infrastructure (left)—internet-accessible, cloud-hosted, and reachable by customers, partners, and threat actors—is governed by a strict ingress-minimization policy. The internal network infrastructure (right) serves the distributed organizational workforce across geographies, devices, and roles, and is governed by progressive trust removal and continuous verification. Both planes converge on a protected resource zone (centre) that is not a network segment but the intersection of application paths, identities, sessions, regulated data stores, and administrative workflows. The organizational boundary (dashed) encloses the protected resource zone and the internal plane, while the external plane is deliberately exposed to the public Internet.

Meanwhile, the adversary landscape facing SaaS operators has intensified. The Verizon DBIR, the CrowdStrike Global Threat Report, and the Cisco Talos Year in Review converge on the same leading breach vectors: credential theft, identity-based attacks, and web application exploitation [10]-[12]—CrowdStrike reports identity-based initial access in the majority of 2024 intrusions, and Talos identifies valid-account abuse as the single most common initial-access technique for the same period [11] [12]. MITRE ATT&CK catalogs hundreds of techniques exploiting exactly the trust assumptions Zero Trust is designed to eliminate: lateral movement through standing credentials, privilege escalation via misconfigured roles, and exfiltration over legitimately provisioned paths [13]. In regulated SaaS environments, the consequences are amplified: the attacker's prize triggers mandatory breach notification, regulatory sanction, and customer harm.

This paper advances a specific, operational interpretation of Zero Trust: the goal is not universal denial, but *minimum standing trust consistent with business-required operations*—abbreviated throughout as Minimum Standing Trust (MST).

We define the two central terms precisely at the outset. By *standing trust* we mean any privilege, session, credential, key, or access path that persists by default beyond the specific transaction that justified it—for example, a never-expiring service-account token, a broadly scoped administrative role that is held continuously rather than acquired on demand, or a session that remains valid without renewed proof of identity, device posture, and context. *Minimum Standing Trust* is the operating objective of holding the aggregate quantity of such persistent, unjustified trust to the smallest value consistent with business-required operations, and of replacing persistence with continuous, evidence-gated re-authorization wherever feasible.

MST is related to, but distinct from, three established principles. *Least privilege* [14] constrains the scope of a grant—how much access an identity holds; MST additionally constrains its *persistence*—how long and under what continuing evidence it holds it. *Zero standing privilege* eliminates durable administrative rights in favour of just-in-time elevation; MST generalizes the idea from administrative privilege to the entire action space, including sessions, service identities, and machine-to-machine paths, and embeds it in a cost model that also prices denial. And *continuous verification* re-evaluates trust signals over time; MST specifies what the re-evaluation must decide—the per-action threshold rule of Equation (9)—and what system-level quantity it is optimizing.

The argument rests on four claims that form the spine of the paper.

First, Zero Trust fails in practice not from insufficient controls but from a category error: practitioners confuse Zero Trust with universal denial, then carve back exceptions until the business can function. The correct objective is not to maximize blocking but to eliminate *standing* trust—privileges, sessions, and access paths that persist beyond the transaction that justified them.

Second, the enforcement logic at the Internet-facing application edge must be fundamentally different from the logic governing the internal enterprise: edge controls belong as close to deny-by-default as the service can tolerate, while internal controls progressively remove standing trust through identity, device posture, session context, and data-aware authorization. WAF, NIDS, and bidirectional reputation-based auto-blocking are edge and perimeter tools, not a strategy; the real message is constrain exposure, constrain privilege, and prove continuously. Network location is not a trust signal. Standing privilege is the primary attack surface.

Third, a unified telemetry plane is the operational hinge of the entire model. Without it, every other control degenerates into isolated noise. With it, the security operations center can answer the decisive questions—who acted, from where, against which resource, with what trust context, and with what consequence—without manual event correlation.

Fourth, agentic AI can safely accelerate triage and response, but only when its autonomy is bounded by policy, reversibility, and blast-radius constraints that the

organization has explicitly authorized. AI should operate inside the MST model, not alongside it as a separate problem domain. The governance criteria that determine whether an access action is permitted are the same criteria that determine whether an automated response action is permissible.

The Zero Trust eXtended (ZTX) ecosystem framework [15] extended Kindervag's network-centric formulation into a full enterprise operating model spanning data, workloads, identities, devices, networks, visibility and analytics, and automation and orchestration. ZTX matters here because it legitimized treating Zero Trust as an operational program with measurable pillar-by-pillar maturity—the framing MST inherits and sharpens into an explicit optimization objective.

Compelling and Innovative Aspects of This Work

This paper advances the Zero Trust literature in five specific ways that, taken together, distinguish it from prior practitioner-oriented and academic treatments.

First, a structural diagnosis of why Zero Trust programs fail in practice. The dominant failure mode—control accumulation without proportional outcome improvement—is explained as the natural consequence of treating security posture as monotone in control count, and the diagnosis is grounded in the documented gap between Zero Trust intent and operational deployment [16].

Second, a formal optimization model with parameters that map directly to operational policy levers. The MST framework introduces binary decision variables x_i over the action tuple $a_i = (\iota_i, \delta_i, \rho_i, \sigma_i, \omega_i)$, a risk-exposure weight c_i, a denial-cost weight d_i, and a resource-sensitive verification threshold $\theta(\rho_i, t)$ tied to a continuous verification score $V(a_i, t)$. Each parameter has a defined operational interpretation (Section 6.2), allowable values, and worked-example calibration ranges, so the model is implementable rather than purely expository.

Third, an honest complexity separation, a per-decision enforcement rule, and an action-space-time graph formulation. We prove the complete-information MST problem is solvable in closed form (Proposition 1) and locate the genuine optimization content where it operationally lives: legitimacy uncertainty, which yields a constant-time expected-cost enforcement rule (Equation (9)), and capacity-coupled planning, which we prove NP-hard by an exact reduction from 0-1 knapsack (Proposition 2) and to which the classical knapsack machinery therefore transfers directly. The graph formulation connects the model to provenance-graph datasets, and the MST optimum coincides with a principled minimum of the enterprise attack surface in the Manadhata-Wing sense.

Fourth, a unified two-plane operational architecture with asymmetric enforcement logic. Edge controls converge on deny-by-default with governed exceptions; internal controls converge on progressive standing-trust removal; and the two planes are joined by a single telemetry and policy layer so that evidence, enforcement, and audit share one record.

Fifth, a unified governance surface for AI-assisted security operations. Bounded-autonomy AI is treated as a policy artifact with explicit rungs, reversibility classes,

and approval tiers, aligned with NIST AI RMF, joint-agency, and OWASP agentic guidance, so that machine scope expands only with demonstrated reliability.

The remainder of the paper puts operational architecture first and formal structure second. Section 2 reviews related work; Section 3 presents the MST model and its edge and enterprise practices; Section 4 develops the unified telemetry plane; Section 5 defines policy-governed autonomous response and the bounded-autonomy ladder; Section 6 formalizes MST as a constrained optimization problem and maps the formal elements onto the architecture; Section 7 states the expected outcomes as testable hypotheses and the validation agenda; Section 8 concludes.

2. Related Work

2.1. Zero Trust: Foundations and Frameworks

The intellectual lineage of Zero Trust begins with the observation that security models grounded in network topology—hard perimeter, soft interior—stopped describing reality as workforces went mobile, workloads moved to cloud, and partners joined the interior. The Saltzer-Schroeder principles of least privilege and complete mediation [14] anticipated the core commitments decades earlier; Zero Trust is, in an important sense, their systematic application to identity-centric, cloud-era enterprises.

The BeyondCorp initiative [3] [4] provided the first large-scale empirical evidence that a Zero Trust model could operate at enterprise scale without destroying productivity, documenting the migration of a global workforce off privileged network segments onto access decisions grounded in device inventory, certificate-based access control, fine-grained per-application proxies, and continuous device and user validation [17].

The Zero Trust eXtended (ZTX) ecosystem framework [15], developed at Forrester, extended Kindervag's architecture by explicitly naming the seven pillars of a mature Zero Trust program—network, data, workloads, devices, people, visibility and analytics, and automation and orchestration—and offering measurable maturity criteria for each. ZTX is directly referenced in the CISA maturity model [7] and is the principal scaffolding on which the present paper's multi-pillar analysis is built. The Zero Trust Networks book [5] synthesized Google's operational experience and prior academic work into a coherent engineering guide. The book's formulation of trust as a continuous, context-sensitive score rather than a binary network-location attribute anticipates the adaptive authorization mechanisms that later NIST guidance would formalize.

2.2. NIST and Policy Normalization

NIST SP 800-207 [1] represented the maturation of Zero Trust from practitioner framework to normative federal guidance, defining the logical components of a ZTA—policy engine, policy administrator, policy enforcement point—and articulating seven tenets any Zero Trust program should exhibit. These tenets are

foundational to the present work. NIST SP 800-207A [6] extended this work into cloud-native and multi-cloud environments, adding application-level service identity, ingress and egress gateway policy, and workload-centric enforcement to the ZTA framework.

The CISA Zero Trust Maturity Model [7] operationalized these tenets into five implementation pillars—identity, devices, networks, applications and workloads, and data—with visibility and analytics, automation and orchestration, and governance as cross-cutting capabilities, and defined staged maturity levels from traditional through optimal. Its significance for this work is the recognition that Zero Trust is achieved incrementally, pillar by pillar, along an assessable progression—an implementation path compatible with the tiered trust-reduction approach the MST framework advocates.

2.3. Zero Trust in Cloud and SaaS Environments

The application of Zero Trust to cloud-native and SaaS architectures raises challenges not fully addressed by perimeter-era frameworks. A 2022 comprehensive survey [18] organized the literature around core ZTA components—trust engines, policy engines, continuous authentication—and identified open challenges directly relevant here: the difficulty of continuous trust evaluation at scale, the absence of standardized trust-scoring approaches, and the operational cost of fine-grained enforcement. This finding directly motivates the unified telemetry plane emphasized by the MST framework.

Early cloud security research [19] explored security service-level agreements (Sec-SLAs) for cloud computing, documenting the difficulty of defining quantitative security metrics and monitoring delivered security levels between providers and consumers—a concern MST addresses directly by expressing posture in explicit, measurable terms: verification thresholds, cost weights, outcome metrics. A systematic literature review of BYOD security [20] found that the issues personal devices introduce span technical, management, and people dimensions and therefore require holistic treatment; MST's device-aware trust tiering operationalizes this conclusion by assigning access scope as a function of verifiable device management state rather than a binary managed/unmanaged policy.

A multivocal literature review [16] found that while Zero Trust is broadly endorsed as an architectural goal, evidence of successful end-to-end implementation in enterprise environments remains sparse, and the gap between policy intent and operational deployment is a recurring theme. The MST framework explicitly addresses this gap by treating the optimization of operational throughput as a first-class design objective alongside security assurance.

2.4. Identity, Behavioral Analytics, and Adaptive Authorization

The shift from network-location trust to identity-centric trust has produced a substantial body of research on adaptive authorization. The role-based access control model [21] remains the most widely deployed authorization paradigm, but its

static role assignments conflict with Zero Trust's demand for continuous re-evaluation; attribute-based and policy-based access control incorporate dynamic attributes—device posture, location, behavioral signals—into authorization decisions, moving closer to the dynamic trust evaluation ZTA prescribes [22].

Behavioral User and Entity Analytics (UEBA) provides a complementary evidence layer: statistical modeling of peer-group norms, session characteristics, temporal patterns, and data-access volume can surface anomalies that evade rule-based controls [23]. The MST framework incorporates behavioral conformance as one input to the verification score rather than as a standalone detection product.

2.5. Security Policy as Optimization

The idea of formulating security policy as a constrained optimization problem has precedent: diverse firewall design has been formalized with correctness and optimization objectives [24], attack graphs have long supported quantitative hardening decisions [25], and the attack-surface literature [26] [27] supplies the measurement semantics that make "smaller surface" a well-defined objective rather than a slogan. MST extends this tradition from static configuration artifacts to the full, time-varying action space of an operating enterprise, formulating Zero Trust policy as a constrained edge-selection problem with an explicit complexity separation (Section 6.4).

A parallel and older tradition concerns *attack surface minimization* as an explicit engineering objective. Howard, Pincus, and Wing [26] introduced the attack surface as a measurable property of a system—the subset of its resources reachable and usable by attackers—and argued that reducing this surface should be a first-order design objective. Manadhata and Wing [27] subsequently formalized this as a quantitative metric based on entry points, exit points, and channel resources, and established it as a peer-reviewed evaluation tool for comparing relative security postures across system versions and configurations. This attack surface tradition connects directly to the MST formulation: the permitted action set $\mathcal{T}$ is the attack surface a successful attacker can exploit, and the trust surplus $\mathcal{T} \setminus \mathcal{L}$ is precisely the avoidable portion of that surface. MST generalises attack surface minimization by introducing the operational-cost constraint that prevents over-minimization.

The present paper builds on both traditions. First, it extends the action space from network flows and firewall rules to the richer tuple space of (identity, device, session, resource, operation) that characterizes modern SaaS access patterns. Second, it introduces the action-space-time graph as a unified representation that captures both spatial access structure and temporal verification dynamics in a single model amenable to graph-theoretic analysis, and formulates Zero Trust policy as a constrained edge-selection problem with an explicit complexity separation (Section 6.4). Third, it integrates reputation-based pre-filtering—operationalized through feeds such as AbuseIP DB [28] and Spamhaus [29]—as a mechanism that reduces the effective action space $\mathcal{S}$ before the optimization problem is formu-

lated, further tightening the achievable minimum.

2.6. AI and Automation in Security Operations

The integration of machine learning and AI into security operations has accelerated substantially. Early work demonstrated deep learning for unsupervised insider-threat detection in structured activity streams [23], and UEBA has since matured into a standard SOC capability. Automated extraction of structured threat actions from unstructured threat intelligence—exemplified by TTPDrill [30]—demonstrated early that natural-language security artifacts can be converted into machine-actionable detection logic, a precursor of today's LLM-assisted enrichment. The current frontier is agentic: LLM-based systems that enrich alerts, summarize cases, and propose or execute response actions, with surveys documenting rapid capability growth alongside novel risks—hallucination in threat reasoning, prompt injection through ingested telemetry, and over-delegation of consequential actions [31] [32]—precisely the risk class the bounded-autonomy ladder of Section 5 governs.

Recent work on LLM-assisted security operations has sharpened these concerns: a recent survey [31] finds that while LLMs substantially accelerate analyst workflows across threat intelligence, vulnerability analysis, and autonomous testing, they introduce new attack surfaces—prompt injection against agentic tools and hallucination-induced false negatives—and a 2024 survey of autonomous LLM agents in SOC environments [32] argues that action boundaries and reversibility constraints are necessary preconditions for safe deployment, directly corroborating the bounded-autonomy ladder formalized in Section 5.

NIST's AI Risk Management Framework [33] and its Generative AI Profile [34] provide a governance vocabulary—govern, map, measure, manage—that maps naturally onto the bounded-autonomy ladder: governing the agent's action space in policy, mapping each action class to its risk and reversibility profile, measuring autonomous-action outcomes against expected behaviour, and managing drift through periodic review. The joint agency guidance on secure AI deployment [35] [36] and the NCSC secure-AI development guidelines [37] add the deployment-security dimensions—provenance of inputs, adversarial robustness, monitoring for model drift—that the telemetry plane of Section 4 is positioned to supply as a byproduct of its normal operation.

2.7. Synthesis and Gap Identification

The extant literature collectively establishes that Zero Trust is architecturally sound, empirically motivated, and institutionally mandated—and simultaneously that its operational realization remains under-specified: surveys document persistent gaps between reference architectures and deployed practice, and multivocal reviews find that outcome measurement is the least developed element of published Zero Trust knowledge [16] [18]. That gap—between architectural prescription and measurable operation—is precisely where this paper positions its contribution.

What MST adds beyond existing frameworks. The novelty of MST is sharpest when stated by direct contrast with the frameworks it builds on. Beyond *NIST SP* 800-207 [1], which specifies the logical components (policy engine, policy administrator, policy enforcement point) and the seven tenets of a Zero Trust architecture but leaves the trust-reduction *target* unquantified, MST supplies an explicit optimization objective—minimize standing trust subject to business completeness and verification sufficiency—that turns those tenets into a measurable quantity. Beyond *NIST SP* 800-207*A* [6], which extends ZTA to cloud-native service identity and ingress/egress gateway policy, MST contributes the asymmetric two-plane result: it shows formally why edge and internal enforcement must operate at structurally different parameter values rather than under one uniform policy. Beyond the *CISA Zero Trust Maturity Model* [7], which orders implementation into pillars and maturity stages but treats greater maturity as monotonically better, MST contributes the U-shaped cost geometry (Section 6.3) that explains why monotonic control accumulation overshoots the optimum and can degrade outcomes. And beyond prior *Zero Trust surveys and multivocal reviews*, which document the persistent gap between Zero Trust intent and operational deployment [16] [18], MST advances from *documenting* the gap to giving it a structural account: the gap is the distance between the realized permitted set $\mathcal{T}^*$ and the business-required minimum $\mathcal{L}$, a quantity the framework makes measurable and ties to the established attack-surface metric [27]. Finally, beyond risk-adaptive and attribute-based authorization (ABAC/PBAC and risk-adaptive access control) [22], which already condition individual access decisions on contextual risk signals, MST contributes three things those schemes leave unspecified: explicit cost semantics for the two ways a decision can be wrong (c_i for wrongful permission, d_i for wrongful denial), a resource-indexed hard verification gate $\theta(\rho_i, t)$ that no cost trade-off can override, and a system-level objective—minimum standing trust—of which the per-decision rule (Equation (9)) is the provably optimal enforcement and against which aggregate posture can be measured and governed. In each case, the prior framework supplies the architecture, or the empirical evidence of the problem; MST supplies the optimization structure that makes the trust-reduction objective explicit, measurable, and plane-differentiated.

3. The Minimum Standing Trust Model: Operational Sketch

The MST framework treats Zero Trust implementation as a constrained optimization problem rather than a policy posture: find the minimum-cost permitted action set $\mathcal{T}$ that covers all business-required operations $\mathcal{L}$ while continuously verifying that each permitted action meets a resource-sensitive evidence threshold Θ. Each action carries a risk-exposure cost c_i (the cost of incorrectly permitting it) and a denial cost d_i (the cost of incorrectly blocking it), yielding a two-term objective that makes explicit the trade-off most Zero Trust programs resolve implicitly and badly. The full formalization—decision variables, constraints, complexity, and the connection to attack-surface minimization—is de-

veloped in Sections 6.1 - 6.6; this section gives the operational gist. The paper presents operational architecture first (Sections 3.1 - 3.4) and the formal framework afterward (Sections 6.1 - 6.6), with Section 6.6 mapping each formal element back onto specific architectural components.

The MST model has three direct operational implications that shape the architecture described in Sections 3 - 5. First, the verification score $V(a_i, t)$ can only be computed accurately if telemetry from all relevant planes is fused—making unified telemetry a correctness requirement, not a convenience. Second, the risk weight c_i and the threshold $\theta(\rho_i)$ differ fundamentally between the exposed application edge and the internal enterprise, which is why symmetric control policy across both planes is structurally wrong. Third, the blast-radius and reversibility criteria used to bound AI autonomous response are derived from the same c_i and θ parameterization: high-consequence resources require human approval before their access policy changes for exactly the same reason they require higher verification thresholds.

3.1. A Two-Plane Operating Architecture

The most useful way to operationalize the MST model is to separate the environment into two enforcement planes with different default postures. The external plane—everything reachable from the Internet—warrants deny-by-default with narrow, well-documented exceptions. The internal plane—the distributed enterprise of people, devices, and administrative workflows—warrants the progressive removal of standing trust rather than blanket denial, because legitimate work there is too heterogeneous to enumerate in advance.

The separation of a defended network into an exposed external zone and a controlled internal zone is not itself novel; it reflects current practice across mature enterprise security programs and is implicit in NIST SP 800-207A's ingress/egress gateway model [6] and in the CISA maturity model's application and network pillars [7]. The contribution of the two-plane architecture in this paper is not the zone separation itself, but its *explicit mapping to the MST optimization structure*: the action space $\mathcal{S}$, the risk weights c_i, the denial costs d_i, and the verification threshold function $\theta(\rho_i, t)$ take on fundamentally different parameter values in the two planes, which is why applying symmetric control policy across both planes is not merely suboptimal but structurally inconsistent with the optimization problem's geometry. The gap between symmetric-policy programs and zone-differentiated programs is consistently documented in Zero Trust implementation reviews [16].

Edge traffic is comparatively easier to constrain because the set of legitimate flows is narrower, more observable, and more closely tied to the application's business purpose. Internal enterprise traffic is more heterogeneous and more tightly coupled to changing work patterns, which makes blanket deny-all policies far more likely to produce operational self-harm. NIST SP 800-207A reinforces this separation by emphasizing granular application-level policies, service identities, ingress and egress gateways, and enforcement that works across hybrid and multi-

cloud environments [6]. CISA's maturity model further organizes implementation around identity, devices, networks, applications and workloads, and data, with visibility and analytics, automation and orchestration, and governance as cross-cutting capabilities [7].

This separation leads to a practical design rule: control density should be highest where exposure and consequence intersect. At the external edge, that means hardening ingress, minimizing accepted traffic, and placing the burden of justification on everything unusual; internally, it means reducing standing privilege, continuously validating identities and devices, and using session-level and behavioral context to decide whether an action remains consistent with the role, asset, and data involved. The bridge between the planes is a unified telemetry and policy layer that turns events from WAF, NIDS, SIEM, EDR, XDR, IAM, VPN or ZTNA, cloud, and data-access systems into a shared operational picture rather than disconnected alarms.

Figure 2 summarizes the proposed operating model. The primary architectural mistake in many Zero Trust programs is applying one uniform posture everywhere: deny-by-default is correct for the exposed edge but fractures the internal enterprise, while progressive trust reduction is correct internally but too permissive for the edge. The two-plane model applies each posture where it belongs and joins them through shared telemetry and policy.

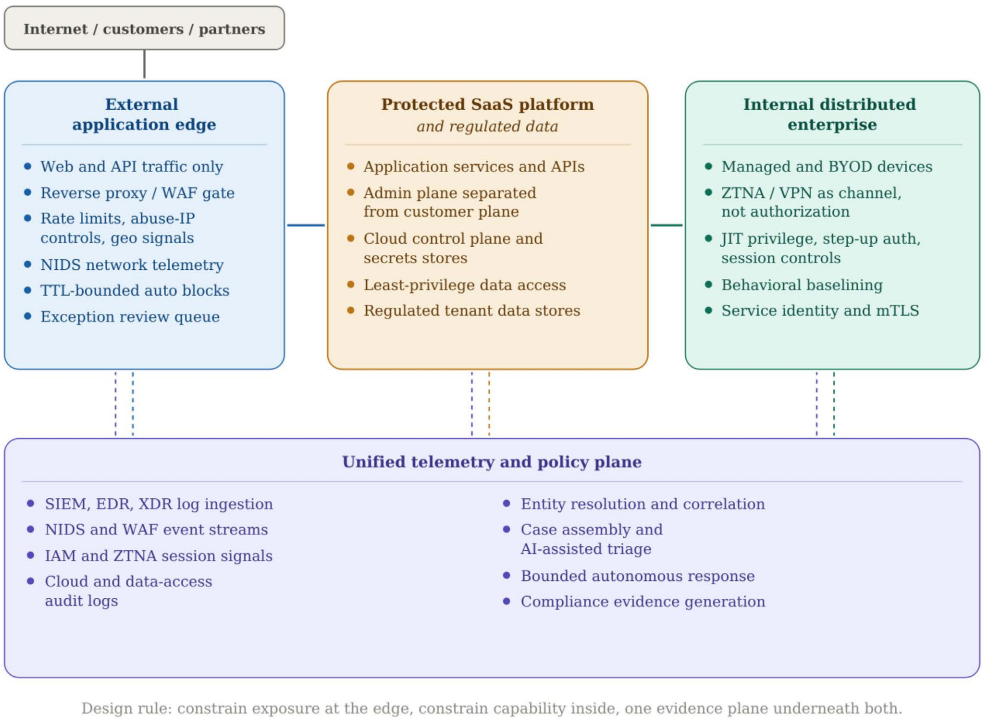


Figure 2. Two-plane operating model for regulated SaaS Zero Trust. The external plane applies deny-by-default pressure at the SaaS edge through tightly constrained ingress, application-aware controls, and documented exception handling. The internal plane progressively removes standing trust through identity, device, session, and data-aware control. Both planes depend on a unified telemetry and policy layer that supports consistent vSOC decision-making.

3.2. Threat Model

The MST framework is designed to address a specific and well-documented adversary profile relevant to regulated SaaS providers. Based on the MITRE ATT&CK framework [13] and annual empirical breach intelligence [10]-[12], the principal threats are:

- **External application attacks**: Web application exploitation (OWASP Top Ten vulnerabilities), API abuse, automated credential stuffing, and bot-driven enumeration. These attacks target the application edge and are best addressed by the external plane controls.
- **Identity and credential compromise**: Phishing, adversarial token theft, session hijacking, and MFA bypass techniques (including adversary-in-the-middle proxies such as Evilginx). These attacks exploit the trust implicit in valid credentials and motivate continuous session re-verification.
- **Privilege escalation and lateral movement**: Exploitation of overprivileged service accounts, standing administrative access, and misconfigured IAM roles. These attacks are addressed by just-in-time privilege, device-trust tiering, and behavioral baselining.
- **Data exfiltration**: Abuse of legitimate data-access paths to extract regulated records, often using credentials obtained in prior stages. This threat motivates data-aware access policy, field-level masking, and egress monitoring.
- **Insider threat and support-account misuse**: Authorized users acting outside normal behavioral baselines, whether through negligence, coercion, or malicious intent. This motivates behavioral analytics and just-enough privilege enforcement.

Command-and-control and known-bad egress: Established footholds—compromised endpoints, rogue servers, hijacked service accounts—that communicate with adversary-controlled infrastructure for command reception, tool staging, or exfiltration. A large fraction of this traffic resolves to IP addresses already catalogued on public and commercial reputation feeds (AbuseIPDB, Spamhaus DROP/EDROP, vendor threat intelligence), motivating symmetric, bidirectional reputation-based blocking at both the network perimeter and the endpoint: inbound connections from known-bad sources and outbound connections to known-bad destinations are both automatically denied.

The MST framework does not claim to defeat all of these threats in isolation. Rather, it provides the architectural substrate within which specific control technologies operate with maximum effectiveness and minimum operational friction.

Trusted inputs and non-goals. MST treats a small set of components as a trusted computing base whose integrity is *assumed* rather than established by the framework itself. Specifically, MST assumes: 1) the identity provider (IdP) and its issuance of authentication assertions are not themselves compromised, so that a validated identity assertion can be relied upon as evidence; 2) the endpoint posture agent (EDR/XDR) reports device state faithfully and has not been subverted on the host it measures; and 3) the unified telemetry pipeline (Section 4) delivers

events with integrity, so that the verification score $V(a_i, t)$ is computed from trustworthy inputs. These three components constitute the framework's root of trust: if an adversary fully controls the IdP, forges device-posture attestations at will, or rewrites telemetry in transit, the verification gate on which MST depends can be defeated and the optimization will operate on corrupted evidence.

Correspondingly, several classes of compromise are explicitly out of scope. MST does not by itself address: supply-chain or fourth-party compromise of components outside the enterprise's direct control (identified as future work in Section 7.7); cryptographic-primitive failure or root-CA compromise underlying the IdP and certificate infrastructure; physical or firmware-level attacks that subvert the posture agent below the layer it can observe; and insider compromise of the governance process that defines the legitimate set $\mathcal{L}$ itself. The defenses for these scenarios—hardware roots of trust, telemetry signing and provenance, IdP-directed threat detection, and separation-of-duties controls on policy change—are complementary measures that MST assumes are supplied by the surrounding security program. The contribution of MST is to minimize standing trust *given* a trustworthy evidence layer, not to guarantee that layer's integrity.

3.3. Application-Edge Enforcement

At the SaaS edge, Zero Trust should be strict: the exposed surface reduced to the smallest practical set of domains, protocols, ports, paths, and partner integrations required for customer use and legitimate operations. If traffic is not necessary for the service to function, it should not reach the service. Administrative interfaces should not be broadly reachable from the public Internet; customer APIs should be versioned and documented; unpublished or legacy endpoints retired rather than hidden; and edge policy should clearly distinguish customer traffic, machine-to-machine traffic, partner integrations, and administrative control paths.

SP 800-207A is especially relevant here because it emphasizes application and service identities in addition to network parameters, enabling a SaaS provider to control not only who is calling an interface but which service is calling which other service and under what policy [6]. This is critical in microservice architectures where east-west service-to-service traffic within the platform can itself represent an attack surface if lateral movement is possible through a compromised service identity.

In practice, the edge control stack—reverse proxy or gateway, WAF, abuse and reputation list enforcement, bot management, rate limiting—fronts the application tier, and the order of operations matters for both efficiency and evidence quality: reputation and abuse-list drops occur first and cheapest, protocol and rate-limit enforcement next, WAF signature and anomaly evaluation after that, and application-level authorization last, so each layer sees only traffic the cheaper layers could not disposition and each drop is logged with the layer that produced it.

Because the legitimate edge surface is relatively compact, autonomous preven-

tion is most realistic at this plane: high-confidence controls can act by default when their scope is narrow and their rollback path simple. The most important such control is bidirectional auto-blocking of IP addresses on curated reputation feeds—AbuseIPDB [28], the Spamhaus DROP/EDROP lists [29], and vendor threat-intelligence bundles—which by construction enumerate hosts with demonstrated adversarial activity; connections to or from them serve no legitimate business purpose. The platform should ingest such feeds continuously and drop both inbound connections from listed IPs and outbound connections to them. The symmetry matters: inbound blocking addresses probing, scanning, and credential stuffing, while outbound blocking denies command-and-control callbacks, tool staging, and exfiltration from any established foothold. Short-lived blocks on exploit-like request patterns and suppression of commodity scanning complement this foundation; geo-based controls can be appropriate when justified by service footprint but are coarse risk signals, not proof of maliciousness. A workable program gives blocked traffic a documented exception path, preserving controlled allowlisting for legitimate business cases.

A further edge consideration is egress discipline from sensitive workloads and administrative enclaves. Providers often invest more in ingress filtering than outbound control, even though exfiltration, command-and-control callbacks, and accidental exposure ride legitimate-looking outbound paths. Zero Trust at the edge therefore benefits from a symmetrical design: published inbound paths should be explicit, but so should the outbound destinations and protocols available to systems handling regulated data or administrative control—service-to-service traffic, update channels, partner APIs, and observability exports enumerated and monitored, everything else measurable and, where feasible, denied by policy.

External Zero Trust also carries a compliance dividend: edge decisions are easier to explain to auditors and customers when the default posture is documented denial with enumerated exceptions, because every allowed path has an owner, a justification, and a review date, and every automated block traces to a named control. The evidence that regulators ask for is generated by the enforcement itself.

3.4. Internal Enterprise Control without Operational Self-Harm

The internal enterprise is where Zero Trust programs most frequently lose credibility. The workforce is distributed across managed corporate endpoints, contractor machines, and BYOD devices acceptable for productivity but not for sensitive administrative access; engineers push code and resolve incidents under time pressure; support staff need constrained visibility into customer-impacting cases; and security administrators need enough reach to investigate and contain. These realities do not invalidate Zero Trust, but they require a more precise implementation model than simple deny-all.

The ZTX framework's seven pillars [15] provide a useful organizing structure for the internal enterprise controls advocated by the MST framework. The following subsections address the four pillars most directly implicated in SecOps oper-

ations for regulated SaaS: identity, devices, data, and behavioral analytics. The remaining pillars—network, workloads, and automation—are addressed in Sections 4 and 5.

3.4.1. Identity as the Primary Policy Object

NIST's Zero Trust guidance removes implicit trust based on network location and instead places authentication and authorization before each session to an enterprise resource [1]. For regulated SaaS operations, this means a support engineer's access to a customer-data console is evaluated per session—identity, device posture, and context verified each time—rather than inherited from being "on the VPN," and the same discipline applies to service-to-service and administrative paths.

The internal question is not simply whether a user reached the corporate network; it is whether this identity, on this device, in this session context, should perform this action against this resource right now. Answering it requires identity assurance, device posture, session freshness, and behavioral conformance evaluated together—the evidence fusion that the unified telemetry plane of Section 4 exists to provide.

A related concern is service-account and machine identity: in regulated SaaS environments, service accounts frequently hold the broadest and longest-lived privileges in the estate, yet receive the least verification. MST treats them as first-class identities—scoped narrowly, rotated automatically, monitored behaviorally, and subject to the same standing-trust reduction as human accounts.

3.4.2. Device-Aware Trust Tiering

BYOD does not need to be eliminated to make Zero Trust credible, but it does require honest segmentation of what those devices are permitted to do, since the security issues personal devices introduce are well documented and span technical and governance dimensions alike [20]. A mature SaaS program can allow low-sensitivity collaboration and basic ticketing from lightly trusted devices while reserving administrative consoles, production shells, sensitive data paths, and broad incident-response capabilities for managed and policy-compliant endpoints. Device posture checks can include verified operating-system patch state, endpoint protection agent health and version, full-disk encryption status, certificate presence, screen-lock policy compliance, and absence of jailbreak or unauthorized administrative tool installation.

The practical effect is that a user may be fully authenticated while the device remains insufficiently trusted for a high-consequence action. This tiering avoids the binary choice between "block all unmanaged devices" and "allow everything." It also provides a defensible audit position: the organization can demonstrate that regulated data paths are accessible only from devices meeting documented posture standards.

3.4.3. Just-in-Time and Just-Enough Privilege

Persistent administrative access is one of the most durable forms of unjustified

trust in modern enterprises: roles granted for a project and never revoked, break-glass accounts that quietly become daily drivers, and admin group memberships that outlive their owners' responsibilities. JIT elevation replaces this stock of standing privilege with a flow of short-lived, evidenced grants tied to a ticket, an on-call rotation, or an approval.

This is where Zero Trust and privacy engineering overlap operationally: the most effective way to protect regulated data is frequently to avoid exposing it at all. Least privilege [14] requires not just that the initial grant be minimal but that it be reviewed and revoked when the business justification lapses; automated JIT systems that provision and expire privilege on verified request—integrated with change-management ticketing, on-call rotations, and data classification—operationalize the principle at the scale and speed a SaaS engineering organization requires.

3.4.4. Behavioral Context and Continuous Verification

Behavioral modeling is compatible with but not identical to Zero Trust: it detects when a nominally authorized identity acts outside its expected pattern, enriching rather than replacing policy-based decisions. Internal baselining can be constructed around role, time of day, application usage, administrative history, data-access volume, source device, network path, and peer-group norms [23]; these signals raise or lower response confidence. Used properly, they are contextual evidence layered onto identity and policy; used poorly, they generate opaque noise that degrades analyst trust in the platform.

The key is to treat behavioral deviation as a modulator of the verification score $V(a)$ in the MST model—a signal that triggers re-verification, step-up authentication, or session scope narrowing—rather than as an independent enforcement mechanism. This integration is well-supported by the UEBA research literature, which consistently finds that behavioral signals have the highest operational value when combined with identity and policy context [23].

3.4.5. Endpoint-Enforced Reputation Blocking

The reputation-based auto-blocking posture described in Section 3.3 for the application edge has a direct symmetric counterpart inside the enterprise, enforced at the endpoint rather than the network perimeter. EDR and XDR agents already observe every outbound connection from every process on managed hosts; integrating the same curated reputation feeds [28] [29] into the EDR policy engine lets the platform block any process from initiating an outbound connection to a known-bad IP or accepting an inbound one. NIST SP 800-94 establishes reputation-list filtering as a standard IDS/IPS capability [38]; applying it at the endpoint closes the visibility gap created by work-from-home patterns, where endpoint traffic bypasses the corporate boundary entirely.

This symmetric posture is structurally important. An adversary with a foothold on an internal endpoint must eventually communicate outward—for command and control, tool retrieval, or exfiltration—and outbound reputation blocking at

the endpoint turns each such attempt into both a denied connection and a high-fidelity detection event tied to a specific process, user, and device. The blocked connection is simultaneously prevention and telemetry: it feeds the UEBA layer as strong evidence of compromise, lowers the affected session's verification score, and can trigger the containment ladder of Section 5. Prevention and detection are not separate programs at this layer; they are the same control observed from two directions.

The control is well-suited to the bounded-autonomy framework (Section 5): signal quality is high (curated, corroborated feeds), the action is narrow (block one address or range), the blast radius is bounded, and the rollback path is simple (unblock plus allowlist exception). It therefore belongs in the autonomous tier, with short TTLs and automatic expiry as additional safety properties.

A useful shorthand emerges from these five principles: internally, Zero Trust is best implemented as the progressive removal of standing trust rather than the elimination of all connectivity. An organization matures when fewer people retain persistent privilege, fewer unmanaged devices touch sensitive paths, fewer sessions continue without renewed proof, fewer anomalous actions go unexplained, and fewer endpoints can reach known-bad infrastructure—a realistic maturity trajectory that accommodates the mobility and device heterogeneity of modern knowledge work.

4. Unified Telemetry and Decisioning

Architecture becomes operations only when telemetry is fused. Most enterprises already operate capable point products—SIEM for aggregation, EDR or XDR for endpoint and cross-domain detection, WAF for application protection, NIDS for network visibility, identity services for authentication and conditional access, cloud consoles, and data-access logs. The problem is not missing tools; it is that each tool tells its own story while the analyst merges the partial narratives manually, under time pressure, against an adversary who is not similarly constrained.

This operational fragmentation is well-documented. Alert fatigue—detection volume exceeding the analyst's capacity for meaningful triage—is among the most cited productivity barriers in SOC research [10]; true positives are lost in the noise, response latency grows, and skilled analysts burn out, while fragmented tooling multiplies the problem by demanding a separate triage pass per console over overlapping evidence.

A practical SecOps design therefore requires a single ingestion and normalization pipeline built around a small set of stable objects: identity, device, session, resource, action, evidence quality, and business consequence. WAF events, NIDS alerts, EDR detections, cloud audit logs, VPN or ZTNA sessions, SaaS application logs, and data-access events are normalized into a common schema so the vSOC can quickly answer the decisive questions: who or what acted, from where, against which resource, using which trust context, with what likely consequence, and with what confidence. Without that normalization, automated response either stays too con-

servative to matter or acts on incomplete evidence with unpredictable outcomes.

Figure 3 illustrates the four-stage pipeline. The dashed boundary on the left groups all nine telemetry sources, making explicit that the architectural problem being solved is not missing tools but the absence of a single ingestion path. The *Decoder* stage handles log normalization and carries the alert rules that are specific to each telemetry source—the logic that knows, for example, what a NIDS rule-fire threshold or an EDR behavioral indicator threshold should look like before it merits analyst attention. The *Alerting* stage applies those rules as a filter gate: only events that cross the relevance threshold proceed to enrichment and SOC notification; all others are logged but not escalated, directly addressing alert fatigue. The *Enrichment* stage then computes the MST formal model variables ($V(a_i,t)$, $\theta(\rho_i,t)$, and c_i) and attaches ATT&CK mappings and policy constraints. This architecture ensures that the analyst receives an enriched incident rather than a raw fragment, and that computational resources are spent only on alerts the decoder has already qualified as relevant.

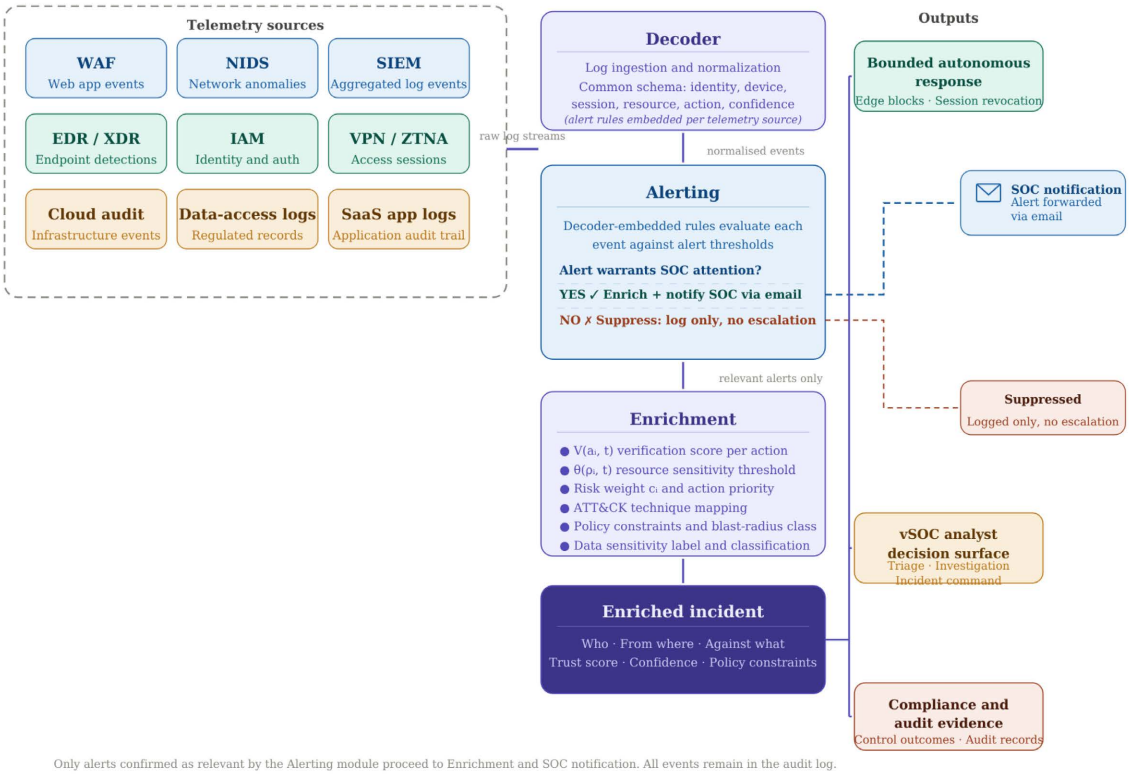


Figure 3. Unified Telemetry & Data Ingestion Pipeline. All nine telemetry sources—WAF, NIDS, SIEM, EDR/XDR, IAM, VPN/ZTNA, cloud audit, data-access logs, and SaaS application logs—are grouped within a single collection boundary and feed a *Decoder* module that performs log ingestion and normalization into a common schema. Decoder-embedded alert rules then drive an *Alerting* module that determines whether each normalized event warrants SOC attention: events that pass are forwarded to the SOC via email notification and proceed to *Enrichment*; events that do not pass are suppressed and retained in the audit log only. The enrichment layer computes the verification score $V(a_i,t)$, applies the resource threshold $\theta(\rho_i,t)$, assigns risk weight c_i, maps adversary techniques, and attaches policy constraints. The resulting enriched incident object is routed to three consumers: bounded autonomous response, the vSOC analyst decision surface, and compliance and audit evidence generation.

4.1. Evidence Classification and Decision Value

To support sound decisions, the normalization layer should classify evidence on two dimensions: technical confidence (how reliable the detection is) and operational consequence (how costly the implied response would be if wrong). The two are independent, and conflating them is a common source of both over-blocking and alert fatigue; keeping them separate lets policy demand high confidence only where consequence is high.

For example, an EDR alert on a kiosk device may be technically high-confidence but operationally low-consequence, warranting immediate autonomous isolation, while the same detection on a production database administrator's workstation is high-consequence and warrants step-up verification and human review before disruptive action—same signal, different policy, because consequence differs.

4.2. Compliance as an Operational Dividend

A unified telemetry plane is also where compliance becomes operational rather than retrospective. Regulated SaaS firms have historically run security and compliance in separate evidence loops—one for incident handling, another for audits, attestations, and customer questionnaires—so analysts resolve events that auditors later need re-explained from a different frame. A shared data plane eliminates this duplication: case notes, control outcomes, policy decisions, and exception records created during operations double as evidence for governance, privacy, and assurance workflows, with appropriate access controls and retention.

This integration supports what the NIST Cybersecurity Framework 2.0 [39] identifies as the "Govern" function: cybersecurity risk management as an organizational capability with defined roles, policies, and oversight—here realized as a property of the operating platform rather than a separate program office.

The design implication bears repeating: unifying telemetry at the dashboard layer improves visualization; unifying it at the decision layer improves operations.[1]

The vSOC should receive *incidents*, not fragments. A suspicious event should arrive with identity context, device trust tier, resource sensitivity, prior related signals, probable adversary technique mapping where applicable, and any policy constraints on what automation is permitted to do next.

5. Policy-Governed Autonomous Response

The rise of agentic AI has revived an old security ambition: a SOC that can not only observe and recommend, but act at machine speed. That ambition is operationally sound only if action is bounded by policy rather than by model confidence alone—and *the policy in question is the MST model itself.* AI governance in secu-

[1]All three authors are affiliated with Athena Security Group. Athena Core [51] and Athena Pallas [52] are commercially available instantiations of the unified telemetry and AI-assisted analyst patterns described in this section. They are cited only as disclosed implementation examples. The framework is intended to be vendor-neutral and is applicable across different tooling environments.

rity operations is not a separate problem domain; it is an instance of the same optimization structure. The permitted action set $\mathcal{T}^*$ that governs human access decisions also governs what an AI agent may do autonomously: if an action would require human authorization when a human requests it, it requires human authorization when an AI agent recommends or executes it.

The governance literature establishes the requirements clearly. NIST's AI Risk Management Framework and its Generative AI Profile require AI to be governed as a risk-management discipline, with trustworthiness, oversight, and measurement built in at design time [33] [34]; the NSA/CISA/FBI joint guidance stresses governance, data provenance, adversarial robustness, and continuous monitoring [35] [36]; and OWASP's agentic-application guidance identifies prompt injection, excessive tool grant, and unsupervised policy modification as the primary attack vectors against SOC-deployed agents [40]. Ferrag *et al.* find that LLMs substantially accelerate analyst workflows while introducing hallucination risk and direct susceptibility to adversarial manipulation of input telemetry [31]; Motlagh *et al.* argue that action boundaries and reversibility constraints are necessary preconditions—not optional preferences—for safe autonomous SOC deployment [32].

The integration of these requirements into the MST model produces a straightforward design rule: *AI agents in security operations should follow the same trust constraints as human analysts, applied to response actions rather than access decisions*. Concretely, an AI agent executing a response action must satisfy the same criteria as any other permitted action: the verification evidence must be sufficient ($V(a_i,t) \geq \theta$), the risk weight must be low (c_i small), and the action must be reversible and time-bounded. This means that an AI agent may block an IP address or trigger step-up authentication autonomously on the same evidence that would justify a human analyst doing so; it may not rewrite access policy, grant durable privilege, or take any action that a human analyst would need approval to take. The MST model thus provides a unified governance surface for both human and AI actors in the security operations workflow.

For cyber defense operations, the safest synthesis is bounded autonomy. An AI assistant provides high value enriching alerts, correlating evidence, drafting case summaries, explaining likely adversary behavior, and recommending next steps; it becomes more sensitive when allowed to suppress noisy alerts, trigger step-up authentication, temporarily isolate a host, revoke a session token, or block an IP range; and it becomes high risk when it can rewrite policy broadly, grant or revoke enduring access, or materially affect customers or regulated data without immediate human review.

5.1. Agentic AI Frameworks for vSOC Operations

The most consequential design decision in deploying agentic AI inside a virtual security operations center is not model selection but the definition of the action boundary: which actions the agent may take autonomously, which require human approval, and which are prohibited outright. MST resolves this with a bounded-

autonomy ladder ordered by blast radius and reversibility. Actions that are narrow, reversible, and pre-authorized—enrichment, deduplication, short-TTL blocks of reputation-listed infrastructure, step-up authentication challenges—sit in the autonomous tier; actions that alter policy, touch regulated data at scale, or cannot be quickly undone sit behind human approval; and open-ended system modification is prohibited regardless of model confidence.

MST provides a principled criterion for this calibration. Each candidate agent action a_i inherits the same parameters that govern human access decisions: a risk-exposure weight c_i, a denial cost d_i, and a resource-sensitive verification threshold $\theta(\rho_i)$ (Section 6.2). The agent's permissible-action set $\mathcal{A} \subseteq \mathcal{T}^*$ is therefore not a property of the agent or the model but of the MST optimum: the agent may act autonomously on actions whose c_i is small, whose verification evidence $V(a_i, t)$ comfortably clears $\theta(\rho_i)$, and whose effect is reversible within an audit window the organization has approved. Every other action is recommended, not executed.

The blast-radius spectrum operationalizes this principle. We define the *blast radius* of action a_i as the magnitude of organizational exposure created if the action is wrong—measured in the same currency as c_i, but extended to capture cascading consequences such as downstream availability impact, customer-facing damage, regulatory disclosure thresholds, and reversibility timeline. At one end of the spectrum sit actions with negligible blast radius: muting a duplicate low-value alert, enriching an incident with reputation context, drafting a triage summary for analyst review. These actions can safely run as fully autonomous provided three structural controls are in place: 1) a strong upstream verification policy that the agent inherits rather than overrides; 2) deterministic action logging into an immutable audit trail; and 3) a kill-switch that an authorized operator can invoke without external dependency. At the opposite end sit actions with severe blast radius: rewriting access policy, granting standing privilege, broad isolation of customer-facing services, irrevocable data destruction. These require explicit human-in-the-loop approval regardless of model confidence, evidence quality, or operational pressure.

Between these endpoints lies the largest and most contested region of the spectrum, where well-governed programs differ legitimately: host isolation, session revocation, credential suspension, and traffic quarantine. These actions are reversible in principle but operationally disruptive when wrong—an isolated production host or a suspended service account interrupts real work. MST's resolution is contextual: the same action class sits at different rungs for different asset classes, with consequence-weighted placement (a kiosk endpoint is not a production database server), short default TTLs, and automatic escalation to human review when the affected asset carries a high c_i or touches regulated data.

Excessive tool grant, identified in OWASP's agentic application security guidance [40], occurs when an agent is configured with more capability than its task requires—write access where read would do, production scope where

staging would do—so that a single compromised or confused agent inherits blast radius its function never needed. Under MST, the agent's tool set is governed like any other identity: scoped minimally, granted JIT where feasible, and audited continuously; excessive grant is a policy violation, not an emergent behaviour.

Adversarial input manipulation, particularly indirect prompt injection embedded in ingested telemetry [40]-[42], deserves a candid treatment, because the attack does *not* require compromising the telemetry pipeline: ordinary telemetry is saturated with attacker-controlled text—usernames, HTTP headers, file paths, log messages, WAF payload excerpts quoted inside alerts—any of which can carry instructions for the language model that later reads the alert, arriving over the normal ingestion path. MST therefore does not claim to prevent prompt injection; it *bounds its consequence* through three structural properties. First, *provenance separation*: $V(a_i, t)$ and c_i are computed by the deterministic enrichment layer (Section 4) from structured evidence fields, never by the language model from narrative text, so injected prose cannot raise the evidence score of the action it advocates. Second, *action-set closure*: the agent's permissible actions are enumerated in policy, so a successful injection can at most trigger an already-preapproved, reversible, low-c_i action—a bounded, auditable nuisance—while higher-blast-radius actions sit behind human approval. Third, *asymmetric thresholds*: autonomous execution requires $V(a_i, t)$ to clear a strictly higher bar than the equivalent human-initiated action, so a fabricated narrative without corroborating structured evidence fails the gate. The residual risk is stated plainly: injection can still trigger nuisance actions within the autonomous tier and can shape the summaries analysts read. Treating telemetry-derived free text as data rather than instructions, recording input provenance for every autonomous action, and red-teaming the injection path (Section 7.4) are mandatory operating practices under MST.

The practical implication for vSOC architects is direct: deploying agentic AI inside an MST-governed environment is not principally a question of model selection but of permissible-action enumeration. The model is the means; the permissible-action set $\mathcal{A}$, the verification thresholds, the reversibility requirements, and the audit instrumentation are the substance. The bounded-autonomy ladder developed in the next subsection makes this substance concrete.

5.2. The Bounded-Autonomy Ladder

The practical test for whether an action belongs on the autonomous side of the ladder is not whether the model is usually right, but whether the action is reversible, narrow, and pre-authorized—and whether the organization can absorb the worst-case wrong execution without material harm. Confidence is an input to the decision; consequence is the boundary.

These criteria explain why temporary edge blocking often qualifies for autonomous execution before permanent access revocation does. A short-lived block on

a source matching multiple abuse signals, triggering a known exploit rule, and affecting no preapproved partner is a more appropriate autonomous action than a long-lived access change against an internal administrator during an active production incident.

The design is also relevant to reinforcement and adaptive learning within the response engine. Statistical models are well-suited to optimizing alert-priority thresholds, ranking playbook choices, learning duplicate-suppression patterns, and calibrating when intervention delay costs more than a reversible action; they are poorly suited to unconstrained self-directed policy modification in production. If learning systems improve autonomous action over time, the reward function must incorporate not only speed and detection gains but also false-block cost, customer impact, action reversals, and exception volume [43]—otherwise the system optimizes the wrong objective.

5.3. Human Factors and SOC Trust in Automation

There is also a human-factors rationale for bounded autonomy documented by the literature on automation-induced complacency and trust miscalibration: analysts working alongside automation calibrate reliance to its observed predictability, and a platform that behaves unpredictably—jumping from alert summarization to disruptive containment without an intelligible policy boundary—breeds either overtrust (rubber-stamping machine output) or undertrust (disabling automation after the first surprise). This is the automation-trust dynamic documented across decades of human-factors research: miscalibrated automation produces disuse after surprises just as reliably as it produces misuse after overtrust [44] [45]. The bounded-autonomy ladder gives analysts a stable, learnable model of what the machine will and will not do on its own, which is what makes calibrated reliance possible.

Figure 4 and Table 1 together illustrate that the safest automation program is one that expands machine speed where reversibility is high and contracts it where policy consequences are broad. Organizations should also resist confusing a more capable model with better policy. An AI assistant can become dramatically more fluent at summarizing incidents without improving the correctness of the response actions the organization has authorized. Conversely, a conservatively bounded policy can remain safe even when the AI layer is only modestly capable, because it limits the damage of incorrect inference. This separation of model capability from policy authority is a fundamental governance principle for agentic security tooling.

Every autonomous action should be explainable in operational terms. The record should show what evidence triggered the action, which policy authorized it, what was done, what the reversal path is, and what happened next—written as control-to-case evidence at execution time. This is what makes automation auditable and what allows the governance cadence of Section 7 to tune policy from outcomes rather than anecdotes.

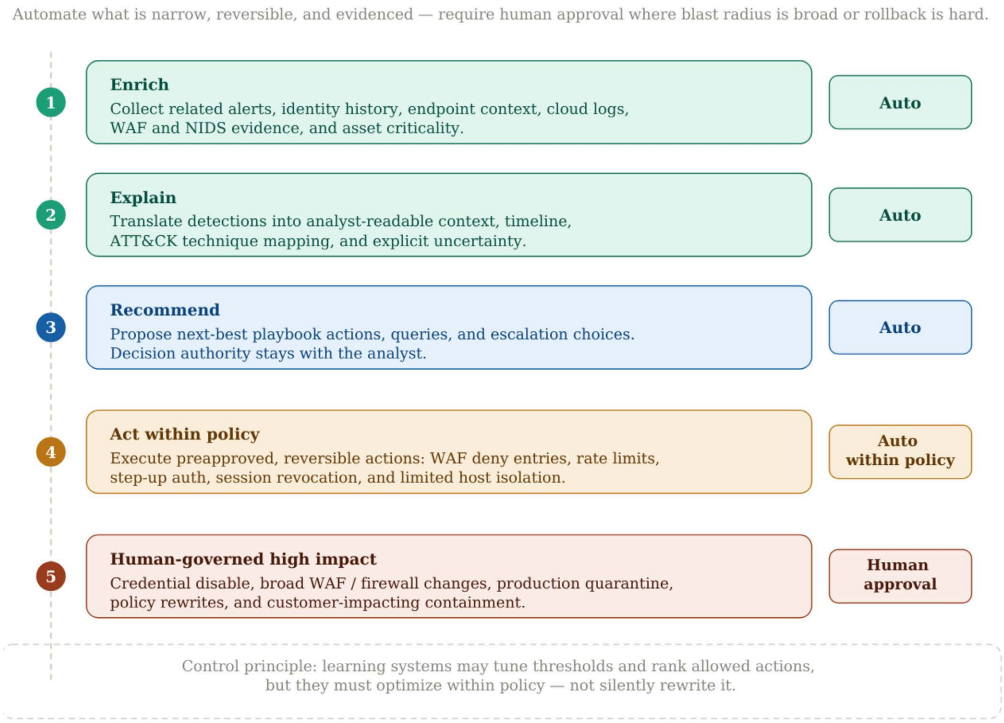


Figure 4. Bounded-autonomy ladder for the vSOC. Low-risk functions such as enrichment, correlation, summarization, and duplicate-noise suppression can be automated aggressively. As blast radius, irreversibility, or customer impact increase, autonomous action should narrow and human approval becomes mandatory.

Table 1. Response decision matrix for policy-governed autonomous response. Signal quality, blast radius, and reversibility together determine whether an action is appropriate for automatic execution or requires human review.

Event class	Typical signal quality	Blast radius/reversibility	Default autonomous action	Human-review trigger
Known-bad Internet source	High confidence from abuse feed plus exploit or scanning context	Low blast radius when block has short TTL and no partner dependency	Temporary IP block or rate limit at edge	Partner traffic, repeated exception requests, or unexpected business impact
Repeated WAF exploit attempts	High when rule precision is validated on the protected route	Low to moderate; reversible at edge controls	Short-lived block, challenge, or request throttling	Possible false positive on customer workflow or API client
Managed endpoint malware	High for confirmed malware or command-and-control indicators	Moderate; rollback available but host isolation affects user productivity	Quarantine artifact; isolate host if policy conditions are met	Executive or on-call operational asset, or uncertainty about detection quality
Duplicate low-value alerts	Medium to high if repetition pattern is stable	Very low; fully reversible	Mute or suppress with expiration and sampling review	Pattern drift, new asset class, or prior false suppression
Privileged session anomaly	Medium; behavior signal strengthened by identity and data context	Moderate to high depending on account and active task	Step-up authentication, token revocation, or narrowed session scope	Critical user, sensitive maintenance window, or customer-visible risk
Broad policy rewrite or permanent access denial	Varies; confidence alone is insufficient	High; often difficult to reverse without operational fallout	No autonomous execution	Always human approval

6. Formalizing the Theoretical Foundations of MST

6.1. Why Zero Trust Implementations Fail as Monotonic Optimization

Before formalizing the MST model, it is worth establishing why the dominant failure mode of Zero Trust programs takes the form it does. Practitioners and academics document a recurrent pattern: programs begin with an architectural goal of radical access reduction, accumulate controls as maturity increases, yet do not proportionally improve measurable security outcomes—Buck *et al.* [16] identify this gap systematically across a multivocal review of enterprise deployments, finding control accumulation without outcome measurement to be the most consistently reported implementation failure.

This failure mode reflects a specific structural error: treating Zero Trust deployment as a monotonic accumulation of controls, where more verification, more segmentation, and more denial are assumed to yield more security. The assumption fails twice. Operationally, every added control consumes finite budgets—user patience, helpdesk capacity, analyst attention, change-window time—so controls added past the operating point displace higher-value enforcement rather than adding protection. Economically, denied legitimate work does not disappear; it re-routes through exceptions, shadow paths, and emergency overrides that are less visible and less governed than the paths they replace. Security posture is therefore not monotone in control count, which is the empirical observation the MST optimization model of Section 6 formalizes.

We are therefore careful to distinguish the MST claim from a simple inversion: the goal is not to *minimize* controls, but to find the *minimum-cost trust set* that satisfies both the security and operational constraints simultaneously. This is a constrained optimization problem, not a greedy minimization.

6.2. Action Space and Decision Variables

Let $\mathcal{S} = \{a_1, a_2, \cdots, a_n\}$ denote the *action space*: the finite set of all access actions possible in the enterprise at a given point in time. Each action $a_i \in \mathcal{S}$ is a tuple:

$$a_i = (\iota_i, \delta_i, \rho_i, \sigma_i, \omega_i) \tag{1}$$

where ι_i is the authenticated identity, δ_i is the device trust tier, ρ_i is the target resource and its data classification, σ_i is the session context (freshness, MFA strength, network path), and ω_i is the requested operation (read, write, execute, elevate, etc.).

Introduce binary decision variables $x_i \in \{0,1\}$ for each $a_i \in \mathcal{S}$:

$$x_i = \begin{cases} 1 & \text{if action } a_i \text{ is permitted } (\text{included in } \mathcal{T}) \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

The *permitted action set* is then $\mathcal{T} = \{a_i \in \mathcal{S} : x_i = 1\}$ and the *legitimate business-required set* $\mathcal{L} \subseteq \mathcal{S}$ is defined as the subset of actions for which a documented business or operational justification exists. Let $\ell_i \in \{0,1\}$ denote membership in $\mathcal{L}$.

Creating and maintaining $\mathcal{L}$ in practice. The legitimate set is a governed register whose entries each carry a justification, an owner, and an expiry. Additions are approved through change management—a resource owner sponsors the request, an access-governance reviewer validates the tuple against job function and data classification, and the entry is recorded against its justifying ticket, supplying the control-to-case traceability of Section 7. Emergency exceptions enter through a break-glass path that grants immediately but attaches a mandatory short expiry and retrospective-review flag, so an emergency grant cannot silently harden into standing trust. Stale entries are removed by *time-based expiry* (every entry must be re-justified or it lapses) and *usage-based reconciliation* (entries the telemetry plane records as unused are surfaced for owner re-attestation or removal). This lifecycle keeps the declared $\mathcal{L}$ aligned with the actions the business actually requires; the residual gap—and the prospect of *learning* $\mathcal{L}$ from observed behaviour—is discussed as a limitation in Section 6.3 and a research direction in Section 7.7.

As a **uniform-cost baseline**, the MST problem in its simplest form seeks the minimum-cardinality permitted set that covers all legitimate actions and satisfies a verification threshold:

$$\mathcal{T}^{*} = \arg\min_{\mathcal{T}} \left|\mathcal{T} \setminus \mathcal{L}\right| \text{ subject to } \mathcal{L} \subseteq \mathcal{T},\ \forall a_i \in \mathcal{T} : V\left(a_i\right) \geq \theta\left(\rho_i\right) \tag{3}$$

where:

- $c_i \geq 0$ is the **risk-exposure weight** for action a_i —the cost of incorrectly permitting it. Allowable values are any non-negative real number. In practice, organisations assign c_i on a relative scale calibrated to data classification and role sensitivity: a read-only query against a non-regulated internal dashboard might be assigned $c_i = 1$; a privileged write to a PHI record store by a service account $c_i = 50$; an administrative action on the cloud control plane by an unmanaged device $c_i = 100$ or higher. The scale itself is arbitrary—what matters is the ratio between values, since the optimiser trades off c_i terms against d_i terms. A practical starting point is a four-tier classification: $c_i \in \{1, 10, 50, 100\}$ mapped to low, medium, high, and critical data sensitivity respectively.
- $d_i \geq 0$ is the **denial-cost weight** for action a_i —the cost of incorrectly blocking it when it is legitimately required. Allowable values are any non-negative real number. For actions that are business-critical and time-sensitive, d_i should be high: blocking an on-call engineer's read access to production logs during an incident might carry $d_i = 200$, reflecting the cost of delayed remediation. For speculative or low-frequency access patterns, d_i may be close to zero. In the uniform-cost baseline (Equation (3)), $d_i \to \infty$ for all legitimate actions, which forces the optimiser to permit every member of $\mathcal{L}$ unconditionally. In the weighted formulation, finite d_i values allow the policy designer to express that some legitimate actions carry acceptable denial risk—useful for soft-blocking patterns that are legitimate but rarely exercised, where temporary disruption is tolerable.

- $\theta(\rho_i) \in [0,1]$ is the **verification threshold** for the target resource ρ_i —the minimum composite evidence score an action must achieve before it may even be considered for permission. Unlike c_i and d_i, which enter the objective function, θ acts as a hard gate: any action with $V(a_i) < \theta(\rho_i)$ is ineligible regardless of its cost weights. Allowable values are in $[0,1]$, where $\theta = 0$ imposes no verification requirement and $\theta = 1$ requires perfect evidence on every dimension—an unachievable bar in practice. A workable implementation maps θ to resource sensitivity class: $\theta = 0.4$ for low-sensitivity internal resources; $\theta = 0.6$ for production systems; $\theta = 0.8$ for regulated data stores; and $\theta = 0.95$ for cloud control plane and secrets management interfaces. The verification score $V(a_i)$ that is compared against $\theta(\rho_i)$ is itself a composite of identity assurance level, device trust tier, session freshness, MFA strength, and behavioural conformance, each component scored in $[0,1]$ and aggregated by policy-defined weighting.

Equation (3) is equivalent to setting $c_i = 1$ and $d_i \to \infty$ for all i in the full formulation: every unjustified permitted action costs equally, and every denied legitimate action is infinitely costly. This uniform version is operationally interpretable—minimize the count of unnecessary permissions—but it ignores the fact that permitting a production-administrator session to a regulated data store carries far greater risk than permitting a read-only dashboard query. Section 6.3 generalises it accordingly.

6.3. Weighted Objective Function

Section 6.2 established that in the uniform-cost baseline, every unjustified permitted action costs equally ($c_i = 1$) and every denied legitimate action is infinitely costly ($d_i \to \infty$). The full formulation replaces those fixed values with action-specific weights, producing the **MST optimization problem**:

$$\min_{x} \sum_{i=1}^{n} c_i x_i (1 - \ell_i) + \sum_{i=1}^{n} d_i (1 - x_i) \ell_i \tag{4}$$

subject to:

$$x_i = 1 \quad \forall a_i : \ell_i = 1 \quad (\text{business completeness}) \tag{5}$$

$$x_i \le \mathbf{1}\left[V(a_i) \ge \theta(\rho_i)\right] \quad \forall a_i \quad (\text{verification sufficiency}) \tag{6}$$

$$x_i \in \{0,1\} \quad \forall i \in \{1, \cdots, n\} \tag{7}$$

Plain-English reading of Equation (4). The equation asks: find the set of permission decisions—each x_i a yes/no on one action—that minimizes the total of two costs: the risk cost of every permitted action the business does not require, plus the denial cost of every required action that is blocked. Constraint (5) says every business-required action must be permitted; Constraint (6) says nothing may be permitted without sufficient verification evidence for its resource class; Constraint (7) says each decision is binary. Under complete information the two cost terms never conflict (Proposition 1); under uncertainty they trade off at every action (Section 6.3.1).

The first term of Equation (4) thus penalizes unnecessary trust: permitting actions that have no business justification ($\ell_i = 0$), weighted by their risk exposure c_i. The second term penalizes operational self-harm: denying actions that are legitimately required ($\ell_i = 1$), weighted by their denial cost d_i. Constraint (5) encodes the business completeness requirement: every action in $\mathcal{L}$ must be permitted. Constraint (6) encodes the continuous verification requirement: an action can be permitted only if its verification score $V(a_i)$ meets or exceeds the resource-specific threshold $\theta(\rho_i)$. The verification score $V(a_i)$ aggregates identity assurance level, device trust tier, session freshness, and behavioral conformance into a composite signal; $\theta(\rho_i)$ is a policy-defined minimum that scales with the sensitivity of the target resource ρ_i.

Policy intent: verification is a hard gate; completeness is conditional on it. Sections 6.2 and 6.3 could be read as supporting two different policies for a legitimate action whose current evidence is insufficient—"always permit" (business completeness) versus "permit only after sufficient evidence" (verification sufficiency). The framework resolves this in favour of verification: *a business-required action is temporarily denied whenever its verification evidence is insufficient*, and permitted only once the evidence clears the resource threshold. The completeness requirement is conditional—the policy must permit every legitimate action whose evidence is *currently* sufficient, not permit legitimate actions unconditionally. When $V < \theta$ for a legitimate action, the correct response is not to waive the threshold but to remediate the evidence gap—step-up authentication, device re-enrolment, session re-establishment—after which the action becomes permissible. No action, legitimate or otherwise, is ever permitted on insufficient evidence.

A worked end-to-end decision. A concrete request grounds the model. An on-call site-reliability engineer (identity ι_i: authenticated via a FIDO2 passkey, identity assurance high) attempts a privileged write (ω_i: write) to a PHI-classified datastore (ρ_i: regulated data, threshold $\theta(\rho_i) = 0.8$) during an active incident. The request arrives from a laptop whose posture agent reports current OS patching and full-disk encryption but a non-compliant screen-lock policy (device tier δ_i: partially compliant), over a session that is MFA-backed but four hours old with no recent re-authentication (session context σ_i: stale). The enrichment layer (Section 4) aggregates the verification components—identity assurance ≈ 0.85, device tier ≈ 0.70, session freshness ≈ 0.55, behavioural conformance ≈ 0.85—into, for this example, $V(a_i, t) = 0.74$. The action is business-required during the incident, so $\ell_i = 1$ and the denial cost d_i is high. But $V(a_i, t) = 0.74 < \theta(\rho_i) = 0.8$, so the verification gate (Constraint 6) is not satisfied; because verification takes precedence, the business-completeness requirement is held in abeyance and the write is *not* granted. The decision is a *temporary deny with a remediation path*: the platform issues a step-up authentication challenge and requires the engineer to bring the device into screen-lock compliance, which raises session freshness and device tier and lifts $V(a_i, t)$ above 0.8. The action is then permitted, the grant is time-bounded to the incident window (a non-standing grant), and the full sequence—evidence components, threshold, decision, and remediation—is written

to the audit log as control-to-case evidence. Had the same request originated from a fully managed device on a fresh, re-authenticated session, $V(a_i,t)$ would have cleared $\theta(\rho_i)$ immediately and the write would have been granted without challenge. This is the MST model operating as intended: business-required access is delivered, but only on sufficient, current evidence, and never as standing trust.

Solution structure under complete information. It is important to be precise about what Equations (4)-(7) do and do not accomplish. When the legitimacy labels ℓ_i are known exactly, the problem is fully separable across actions and admits a closed-form solution.

Proposition 1 (Decomposability under complete information). *Suppose every action* a_i *with* $\ell_i = 1$ *satisfies* $V(a_i) \geq \theta(\rho_i)$, *so that the instance is feasible. Then an optimal solution of* Equations (4)-(7) *is* $x_i^* = \ell_i$ *for all* i, *the optimal objective value is* 0, *and this solution is unique whenever* $c_i > 0$ *for all* $a_i \notin \mathcal{L}$.

Proof. Constraint (5) fixes $x_i = 1$ for every action with $\ell_i = 1$, and the feasibility assumption makes those assignments admissible under Constraint (6); each such action contributes $c_i x_i (1-\ell_i) = 0$ and $d_i (1-x_i)\ell_i = 0$ to the objective. For every action with $\ell_i = 0$, the only objective contribution is $c_i x_i \geq 0$, which is minimized by $x_i = 0$, an assignment that always satisfies Constraint (6). Because the objective is a sum of per-action terms and the constraints do not couple distinct actions, these per-action minima compose into a global minimum with value 0; uniqueness follows since setting $x_i = 1$ for any non-required action with $c_i > 0$ strictly increases the objective.

Proposition 1 should be read as a statement about the *role* of the deterministic formulation: under complete information, Equations (4)-(7) are a formal *specification of the target policy*—permit exactly the verified business-required set—not a computationally interesting search problem, and no interior optimum exists on the deterministic decision variables. If a required action fails the verification gate, the instance as stated is infeasible; the verification-precedence policy resolves this operationally—the action is temporarily denied, at explicit cost d_i, until the evidence gap is remediated. The genuine optimization content of MST enters through two operationally unavoidable departures from this idealization: *legitimacy uncertainty*, developed immediately below, and *capacity coupling* across actions, developed in Section 6.5, where computational hardness genuinely arises.

6.3.1. Legitimacy Uncertainty and the Expected-Cost Decision Rule

The governed register that operationalizes $\mathcal{L}$ (Section 6.2) is always an approximation: declared entries lag reality, emergency paths go unrecorded, and stale entries persist until reconciliation. The policy engine therefore does not observe ℓ_i; it observes evidence. Let $p_i = \Pr(\ell_i = 1 \mid E_t) \in [0,1]$ denote the calibrated probability, given the telemetry evidence E_t at decision time, that action a_i is legitimately business-required—assembled from the register itself, usage history, peer-group access patterns, and the UEBA evidence layer. Replacing the known labels in Equation (4) with this posterior yields the expected-cost objective:

$$\min_x \sum_{i=1}^{n}\left[c_i x_i\left(1-p_i\right)+d_i\left(1-x_i\right)p_i\right] \text{ s.t. } x_i \le \mathbf{1}\left[V\left(a_i,t\right)\ge\theta\left(\rho_i,t\right)\right],\ x_i\in\{0,1\} \quad (8)$$

The hard completeness constraint is here absorbed into the objective—an action believed to be required can be denied, but only at expected cost $d_i p_i$—exactly the operational reality the deterministic model idealizes away. Under Equation (8) the two cost families trade off at every action: permitting a_i risks c_i with probability $(1 - p_i)$; denying it risks d_i with probability p_i. Because the objective remains separable, the optimal policy is the per-action likelihood threshold

$$x_i^* = 1 \Leftrightarrow p_i \ge c_i/\left(c_i+d_i\right) \text{ and } V\left(a_i,t\right)\ge\theta\left(\rho_i,t\right) \quad (9)$$

obtained by directly comparing the two expected costs. Equation (9) is the MST *enforcement* rule: computable in constant time per decision—which makes continuous per-access re-authorization feasible at production request rates—and it makes the weights' roles concrete: raising c_i raises the evidence bar for permitting a_i; raising d_i lowers it. The rule also locates MST relative to risk-adaptive access control in the ABAC/PBAC tradition [22]: where risk-adaptive schemes leave the risk-versus-need trade-off unspecified, MST supplies explicit cost semantics (c_i, d_i), a resource-indexed hard verification gate ($\theta(\rho_i, t)$), and a system-level objective that the per-decision rule provably optimizes.

The operating-point geometry. Organizations tune a small number of global policy levers, not n separate thresholds. Consider the one-parameter family indexed by permissiveness $\tau\in[0,1]$: $x(\tau)$ permits a gated action exactly when its posterior legitimacy is among the highest τ -fraction of the action space, from full denial ($\tau=0$) to permitting every gated action ($\tau=1$). Along this family the expected exposure cost $\sum_i c_i x_i(\tau)(1-p_i)$ is nondecreasing and the expected denial cost $\sum_i d_i\left(1-x_i(\tau)\right)p_i$ is nonincreasing; their sum attains its minimum at an interior operating point τ^* under a simple sufficient condition: the highest-legitimacy actions are more costly to deny than to permit ($d_i p_i > c_i(1-p_i)$) while the lowest-legitimacy actions are more costly to permit than to deny ($c_i(1-p_i) > d_i p_i$). Both hold in any realistic enterprise. The per-action rule of Equation (9) achieves expected cost no worse than the best member of this family; the family is what governance teams sweep when calibrating global posture.

Figure 5 visualises this geometry. The U-shape of the total expected cost is the central reason that adding controls monotonically does not yield monotonic security improvement: left of τ^*, additional denial raises expected denial cost faster than it reduces exposure; every unit of standing trust granted right of τ^* adds exposure faster than it removes friction. Identifying the operating point—and, where telemetry supports it, replacing the global lever with the per-action rule of Equation (9)—is the explicit goal of the MST framework.

Limitations of the formulation. Several limitations of Equation (4) deserve explicit acknowledgement.

Action independence. The formulation treats each action a_i as independent: the cost of permitting or denying a_i does not depend on which other actions are simultaneously permitted. In practice, actions are not independent. Permitting a

read on a regulated dataset and permitting an export to an external endpoint together create an exfiltration path that neither action creates alone. The independent-action assumption is the most consequential modelling simplification in the formulation, and extending the model to capture combinatorial action interactions—at the cost of substantially increased complexity—is an important direction for future work.

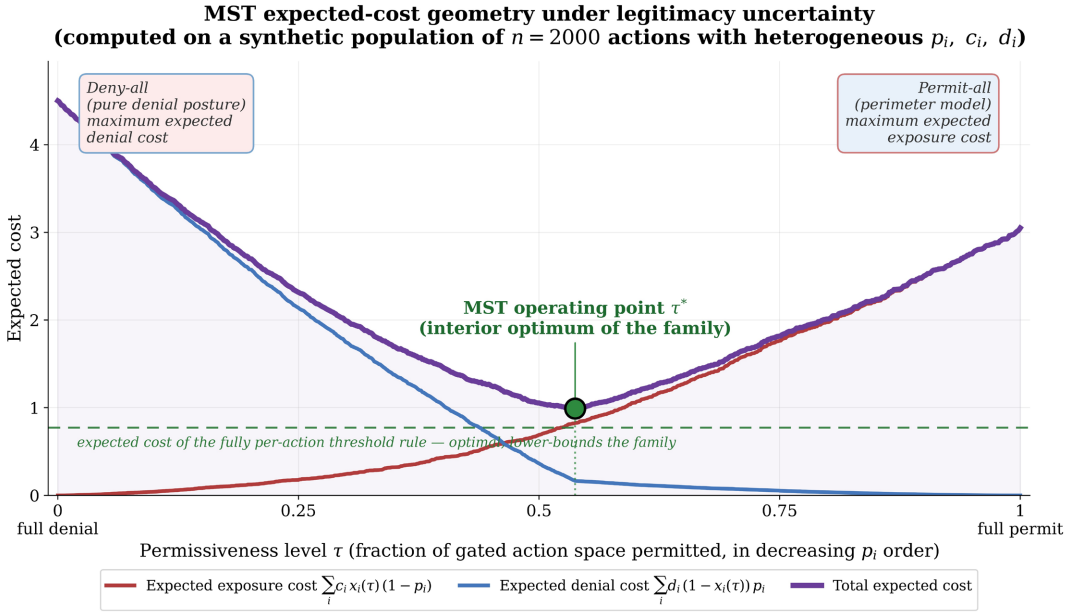


Figure 5. The MST expected-cost geometry under legitimacy uncertainty (Equation (8)), computed on a synthetic population of $n = 2000$ actions with heterogeneous (p_i, c_i, d_i). The horizontal axis is the permissiveness level τ of the one-parameter policy family $x(\tau)$. Expected exposure cost is nondecreasing in τ; expected denial cost is nonincreasing; their sum is U-shaped with an interior minimum at τ^*. The dashed line is the expected cost of the fully per-action rule of Equation (9), which is optimal and lower-bounds every member of the family. Deny-all (left edge) and permit-all (right edge) are both rejected. This geometry arises from legitimacy *uncertainty*: under complete information the optimum is the boundary solution of Proposition 1.

Uniform action weight. Each action is assigned a single scalar c_i, treating all consequences of incorrectly permitting it as commensurable. In practice, an action can have multiple consequence dimensions (regulatory, financial, reputational, operational) that may not reduce to a single number without arbitrary weighting choices. The c_i parameterisation is therefore an abstraction that organisations must calibrate carefully, and different calibrations will produce materially different optimal policies.

The proxy hypothesis. Equation (4) optimises a measurable cost function as a proxy for unmeasurable quantities such as $P(\text{breach})$ or expected breach impact. The hypothesis that minimising the weighted trust surplus reduces actual breach probability is operationally motivated but not proven. It is the foundational modelling assumption of the MST framework, not a derived result, and its empirical validation is identified in Section 7.7 as a priority research direction.

Imperfect knowledge of $\mathcal{L}$. The deterministic formulation treats the business-

required set as known and fixed; in practice $\mathcal{L}$ is dynamic, incompletely specified, and contested. The expected-cost model of Equation (8) addresses this by replacing labels with the posterior p_i, and the graph formulation of Section 6.4 indexes $\mathcal{L}^{(t)}$ over time; what neither resolves is the harder problem of *learning* p_i from operational behaviour rather than policy declarations, identified as a research direction in Section 7.7.

6.4. Action-Space-Time Graph Formulation

The weighted ILP formulation of Equations (4)-(7) treats the action space as a static flat set. In practice, enterprise access patterns are dynamic: identities, devices, sessions, and resources change state continuously, and the verification score $V(a_i,t)$ and threshold $\theta(\rho_i,t)$ are time-varying quantities. A graph-theoretic formulation captures this temporal dimension naturally.

Define the **action-space-time graph** $G=(V_G,E_G)$ as follows. The vertex set $V_G=\mathcal{U}\cup\mathcal{R}$ partitions into *actor nodes* $\mathcal{U}$ (users, service accounts, processes, devices) and *resource nodes* $\mathcal{R}$ (data stores, APIs, administrative consoles, cloud control plane endpoints). Each directed edge $e_{ij}\in E_G$ from actor node $u_i\in\mathcal{U}$ to resource node $r_j\in\mathcal{R}$ represents a potential access action, with weight c_{ij} encoding risk exposure. Time-indexed copies of the graph, $G^{(t)}=\left(V_G,E_G^{(t)}\right)$, capture the fact that verification scores and business requirements evolve: an edge e_{ij} that is permitted at time t may not satisfy the verification threshold at time $t+\Delta t$ if device posture degrades or session context changes.

The MST problem on the graph is: find the minimum-cost subgraph $G^*=(V_G,E^*)$ of $G^{(t)}$ such that:

1) E^* contains all edges corresponding to $\mathcal{L}^{(t)}$ (the current legitimate business-required actions);

2) For every edge $e_{ij}\in E^*$: $V(e_{ij},t)\geq\theta(r_j,t)$;

3) The induced subgraph on $\mathcal{L}^{(t)}$ is connected in the sense that all business workflows can execute without interruption.

The objective function on the graph is $\min_{E^*}\sum c_{ij}$ over permitted edges $e_{ij}\in E^*\setminus\mathcal{L}^{(t)}$, *i.e.*, minimize the total risk-exposure weight of permitted edges that are not legitimately required; when $c_{ij}=1$ for all edges this reduces to Equation (3). We refer to this constrained edge-selection problem as the **MST edge-selection problem**: select a minimum-cost edge set that contains all required edges, subject to per-edge verification gates and, in the coupled variant of Section 6.5, a shared capacity budget. Consistent with Proposition 1, the complete-information instance is solved exactly by $E^*=\left\{e_{ij}\in\mathcal{L}^{(t)}:V(e_{ij},t)\geq\theta(r_j,t)\right\}$. The formulation's value is not that the static instance is computationally hard—it is not—but that it 1) makes time explicit through the indexed copies $G^{(t)}$, turning policy maintenance under evidence drift into an incremental-update problem; 2) is the natural substrate for the uncertainty-weighted and capacity-coupled variants; and 3) maps directly onto provenance-graph datasets for empirical instantiation.

This graph representation, visually depicted in **Figure 6**, connects directly to

intrusion detection systems that model enterprise activity as a provenance graph or process-lineage graph [46]. In those systems, nodes represent processes, files, and network sockets, and edges represent system-call-level events—precisely the fine granularity of the action space described here. The MST edge-selection problem can therefore be instantiated on a provenance graph, treating each system-level event as a candidate x_i and applying the verification and risk constraints per edge. This connection makes the MST formulation amenable to empirical validation on provenance graph datasets, a direction identified as a priority in Section 7.7.

6.5. Complexity and Tractability

The complexity picture for MST is best stated in three layers, in increasing order of operational realism; being precise about which layer is easy and which is hard is itself part of the contribution.

Layer 1: complete information—polynomial. By Proposition 1, Equations (4)-(7) are solved in closed form in $O(n)$ time (plus the cost of evaluating verification scores): permit exactly the verified business-required actions.

Layer 2: legitimacy uncertainty—polynomial. Equation (8) remains separable and is solved exactly by the threshold rule of Equation (9) in $O(1)$ time per decision. This tractability is a design requirement: enforcement-time decisions must be cheap or continuous re-authorization is infeasible. The computational load lives in computing $V(a_i,t)$ and p_i from fused telemetry, not in the optimization.

Layer 3: capacity coupling—NP-hard. Combinatorial hardness enters when permissions are coupled by shared, finite resources: step-up authentication consumes user time and helpdesk capacity, JIT elevation consumes approval throughput, continuous verification consumes telemetry-plane compute, and exception review consumes analyst attention. Let $w_i \geq 0$ denote the capacity consumed by maintaining action a_i in the permitted set during a planning window, and B the capacity available. Restricting attention to business-required actions, the **capacity-coupled MST problem** asks which of them receive the capacity-consuming verified grant now, minimizing the total denial cost of those that must wait:

$$\min_x \sum_{i:\ell(i)=1} d_i(1-x_i) \text{ s.t.} \sum_{i:\ell(i)=1} w_i x_i \leq B,\ x_i \leq \mathbf{1}\left[V(a_i,t) \geq \theta(\rho_i,t)\right],\ x_i \in \{0,1\} \quad (10)$$

Proposition 2 (NP-hardness of capacity-coupled MST). *The capacity-coupled MST problem of* Equation (10) *is NP-hard.*

Proof. By reduction from 0-1 knapsack. Given items with values $v_i > 0$, weights $w_i > 0$, and capacity W, construct an instance with one required action per item ($\ell_i = 1$), $d_i = v_i$, capacity consumption w_i, $B = W$, and $\theta(\rho_i) = 0$. Since $\sum_i d_i(1-x_i) = \sum_i v_i - \sum_i v_i x_i$ and $\sum_i v_i$ is constant, minimizing denial cost subject to $\sum_i w_i x_i \leq B$ is equivalent to maximizing $\sum_i v_i x_i$ under the same constraint—exactly 0-1 knapsack. A polynomial-time algorithm for Equation (10)

would therefore solve 0-1 knapsack, which is NP-hard [47].

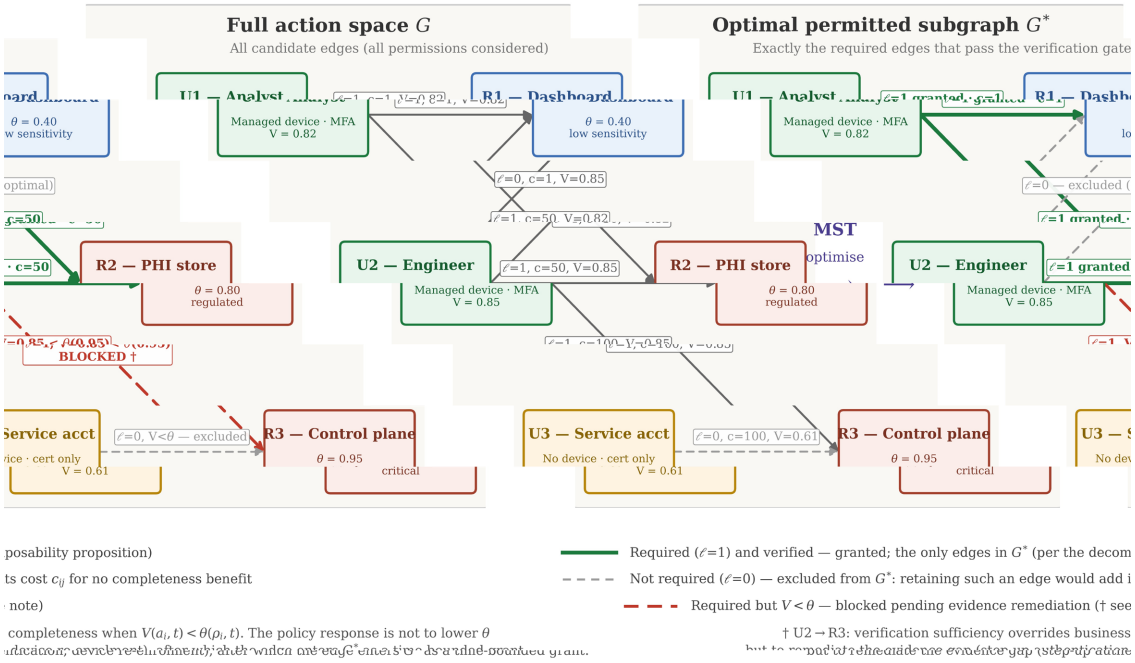


Figure 6. The MST edge-selection problem: full action space G and optimal permitted subgraph G^*. Left: the full bipartite graph of candidate access actions between actor nodes ($\mathcal{U}$) and resource nodes ($\mathcal{R}$), each edge labelled with risk weight c_{ij} and verification score V. Right: the MST-optimal permitted subgraph G^*, which—per Proposition 1—contains *exactly* the business-required edges that pass the verification gate. The grey dashed edge (U2→R1) is eligible but not required, and is excluded: retaining it would add risk cost $c=1$ for no completeness benefit. The key case is U2→R3: the Engineer requires control-plane access ($\ell=1$) but achieves only $V=0.85<\theta=0.95$, so the edge is blocked (red dashed) pending evidence remediation—step-up authentication or device re-enrolment—after which it enters G^*.

Three qualifications keep this result honest. First, 0-1 knapsack is NP-hard only in the *weak* sense: it admits a pseudo-polynomial dynamic program ($O(nB)$ for integer capacities), an FPTAS, and a density-greedy rule guaranteeing at least half the optimal value [47] [48]. Because the reduction of Proposition 2 is an exact correspondence, these results transfer directly, with one precision worth recording: the dynamic program is *exact* and solves the denial-minimization problem outright, while the FPTAS and greedy guarantees are stated for the complementary maximization objective—total denial cost *avoided*—and multiplicative approximation ratios are not, in general, preserved when an objective is complemented. For planning the distinction is immaterial (the planner needs the allocation, and the dynamic program or an integer-programming solver supplies it exactly), but the guarantees should be attributed to the objective for which they actually hold. Second, hardness attaches to the capacity-coupled *planning* problem, not per-request enforcement: the threshold rule of Equation (9) remains constant-time regardless of the budget allocation. Third, at enterprise scale the exact problem is routinely handled by branch-and-bound mixed integer programming solvers, for which the bipartite, separable structure is favourable [49]. The practical

division of labour follows: solve capacity-coupled planning at governance cadence; enforce per-request with the threshold rule.

The time-varying graph formulation introduces additional structure: as $\theta(r_j,t)$ and $V(e_{ij},t)$ change, previously permitted edges may need to be revoked and previously denied edges may become admissible. The dynamic version of the problem is related to *online subgraph maintenance* under changing constraints, a problem studied in the streaming graph algorithms literature [50]. This connection suggests that the SOC's continuous re-evaluation of session trust is not merely a policy preference but a computational necessity: the optimal permitted set changes as the verification evidence changes, and maintaining it requires incremental update rather than full re-solve.

6.6. Mapping Optimization Elements to Architecture

The formalization above maps directly onto the two-plane operating architecture and unified telemetry framework described in Sections 3-5, as summarized below.

Action space $\mathcal{S}$**.** At the external application edge, the action space is narrow: inbound HTTP/S requests, API calls, authentication attempts, and partner integrations. The cardinality of $\mathcal{L}$ at the edge is small and well-documented, making the optimization tractable and the verification threshold θ enforceable close to deny-by-default. Internally, $|\mathcal{S}|$ is orders of magnitude larger (every user-to-resource path across the enterprise), but the risk weights c_i concentrate on a small subset of high-consequence actions (privileged administrative sessions, regulated data access, cloud control plane operations), allowing the optimizer to allocate verification resources proportionally.

Verification score $V(a_i,t)$**.** The unified telemetry plane (Section 4) exists precisely to compute $V(a_i,t)$ in real time. Each source—IAM, EDR, ZTNA, NIDS, UEBA—contributes a component of the verification score. Without telemetry fusion, $V(a_i,t)$ cannot be computed accurately, and the optimization problem becomes infeasible: actions are permitted or denied on incomplete evidence, which corresponds to setting some c_i to zero (under-penalizing risky actions) or some d_i to zero (under-protecting legitimate ones).

Bounded autonomy as a constraint on c_i**.** The bounded-autonomy ladder (Section 5) is a policy translation of the risk-weight structure. Actions at the top of the ladder (policy rewrites, permanent access changes) have high c_i and are therefore excluded from autonomous execution even when $V(a_i,t)\geq\theta$. Actions at the bottom (alert enrichment, duplicate suppression) have negligible c_i and are freely automated. The ladder makes the optimization structure operationally legible to analysts who do not interact with the formal model directly.

Discussion of assumptions. The MST model rests on the working hypothesis that minimizing the risk-weighted unnecessary trust set $\sum_i c_i x_i(1-\ell_i)$ is a useful proxy for minimizing concrete security metrics such as $P(\text{breach})$ or expected breach impact. This hypothesis is not proven in the paper; it is the fundamental modeling assumption, analogous to the assumption in network hardening

that minimizing attack graph reach reduces breach probability [25]. The advantage of the set-size and graph-size formulations is that they are computable from operational data, whereas $P(\text{breach})$ is not. We acknowledge this approximation and identify its empirical validation as a priority research direction (Section 7.7).

6.7. Attack Surface Minimization as the Theoretical Objective

The formulation of Equations (3)-(7) has a direct interpretation in the attack-surface-minimization literature introduced by work on attack surface metrics [26] [27]. In that tradition, the *attack surface* of a system is the subset of its resources reachable and usable by an attacker: entry points through which inputs flow, exit points through which outputs flow, and channel resources that mediate communication. Reducing this surface reduces the opportunities available for adversarial exploitation.

For the MST framework, the connection is immediate. The permitted action set $\mathcal{T} \subseteq \mathcal{S}$ *is* the enterprise attack surface in the relevant sense: every action permitted by policy is an action an attacker might exploit if they can impersonate a legitimate actor or compromise the verification context. The legitimate business-required set $\mathcal{L} \subseteq \mathcal{T}$ is the irreducible lower bound on this surface—the portion that must be exposed for the service to function—and the trust surplus $\mathcal{T} \setminus \mathcal{L}$ is the avoidable portion. The MST objective is therefore a direct quantitative expression of attack surface minimization under operational feasibility constraints.

The effect of reputation-based blocking. A reputation-list control—whether operated at the network perimeter (NIDS, firewall) or at the endpoint (EDR egress policy)—modifies the problem structure by pre-filtering the action space itself, not merely the permitted subset. Let $\mathcal{B} \subseteq \mathcal{S}$ denote the set of actions whose identity tuple $a_i = (\iota_i, \delta_i, \rho_i, \sigma_i, \omega_i)$ has either ι_i or ρ_i resolving to an IP address on a curated reputation feed. Reputation-based auto-blocking enforces $x_i = 0$ for every $a_i \in \mathcal{B}$ regardless of all other considerations. The effective action space becomes:

$$\mathcal{S}_{\text{eff}} = \mathcal{S} \setminus \mathcal{B} \tag{11}$$

and the MST optimization is solved over $\mathcal{S}_{\text{eff}}$ rather than $\mathcal{S}$. Because reputation feeds explicitly enumerate hosts with documented adversarial activity, the expected intersection $\mathcal{L} \cap \mathcal{B}$ is close to empty—legitimate business counterparties do not appear on Spamhaus DROP or AbuseIPDB—so the completeness constraint $\mathcal{L} \subseteq \mathcal{T}$ remains satisfiable in $\mathcal{S}_{\text{eff}}$. The practical result is that the optimal permitted set $\mathcal{T}^*_{\text{eff}}$ computed over $\mathcal{S}_{\text{eff}}$ has an attacker-exploitable surface no larger than that of the corresponding optimum $\mathcal{T}^*$ computed over the full $\mathcal{S}$:

$$\left|\mathcal{T}^*_{\text{eff}} \setminus \mathcal{L}\right| \leq \left|\mathcal{T}^* \setminus \mathcal{L}\right| \tag{12}$$

with strict reduction whenever $\mathcal{B}$ contains any action that would otherwise satisfy the verification threshold under $\mathcal{T}^*$.

Two caveats bound this claim. First, reputation filtering removes only the adversarial infrastructure that is *currently enumerated*: attackers rotate infrastructure faster than feeds update and increasingly stage operations on legitimate hosting, CDN, and residential-proxy space that cannot be blanket-listed, so $\mathcal{B}$ under-approximates the adversarial action space at every instant. The correct reading of Equation (12) is therefore economic rather than absolute: pre-filtering removes the lowest-cost, already-attributed attack paths and raises the adversary's infrastructure cost; it does not shrink the surface available to an adversary willing to pay for clean infrastructure, and it is no substitute for the identity, device, and session controls governing the remaining action space. Second, the near-emptiness of $\mathcal{L} \cap \mathcal{B}$ is an empirical property of feed curation, not a theorem: false positives on shared hosting do occur, which is why the governed exception-review path (Section 3.3) is integral to the control.

The theoretical lower bound. Equation (12) frames the fundamental question of how far attack surface can be reduced in principle. The global lower bound is given by the legitimate set itself:

$$\left|\mathcal{T}^* \setminus \mathcal{L}\right| \geq 0 \tag{13}$$

with equality if and only if every action in $\mathcal{T}^*$ is business-required. This bound is achievable only under perfect information—complete knowledge of $\mathcal{L}$, perfect verification ($V(a_i) = 1$ for all legitimate actions), and no external dependencies. In practice, the realized trust surplus is strictly positive, and the engineering question becomes: how close to zero can it be driven without violating the operational-cost constraint?

The two planes of the MST architecture give different answers to this question. At the external application edge, $\mathcal{L}$ is narrow, well-documented, and relatively static (customer API calls, authentication endpoints, partner integrations), making the trust surplus easier to drive toward zero. Bidirectional reputation blocking reduces it further by shrinking the action space before the optimization is even formulated. Inside the internal enterprise, $\mathcal{L}$ is broader, dynamic, and incompletely specified, so the achievable minimum is inherently larger—but endpoint-enforced reputation blocking still reduces it by an amount bounded by the size of $\mathcal{B}$ restricted to internal-origin traffic.

The MST model thus provides both an operational framework and a theoretical minimum: the ideal end-state of a Zero Trust program is a permitted action set exactly equal to the current legitimate business-required set, with all other actions either denied by policy or pre-filtered by reputation-based blocking. Every deviation from this ideal—every permitted action outside $\mathcal{L}$, every connection to or from a reputation-listed IP that the platform fails to block—is a quantifiable gap in the architecture rather than an unmeasurable policy judgement. This quantifiability is what distinguishes attack surface minimization from traditional Zero Trust rhetoric: it supplies a target that organizations can measure progress against, and it supplies a theoretical justification for specific controls (reputation blocking, JIT privilege, device tiering) in terms of the surface-reduction

they produce.

7. Expected Outcomes and Argument Summary

7.1. Summary of the Central Argument

The argument of this paper can be stated plainly. Zero Trust is not a product category. Network location is not a trust signal. Standing privilege is the enemy. Compliance evidence is a byproduct of good architecture, not its purpose. These are not marketing slogans; they are operational commitments with specific engineering consequences.

The Minimum Standing Trust model operationalizes these commitments in four steps. Step one: at the Internet-facing edge, enforce deny-by-default, with WAF, NIDS, and abuse-list controls as instruments of that posture—not the posture itself. Step two: inside the enterprise, eliminate standing privilege through identity-centric access, device-trust tiering, JIT elevation, and continuous session verification, so no adversary who acquires a credential inherits the access it once carried indefinitely. Step three: fuse all telemetry into a single decision plane so the SOC receives incidents, not fragments, and can answer who acted, from where, against what, with what evidence. Step four: let AI operate inside this model—automating what is reversible, narrow, and pre-authorized while requiring human judgment for anything that cannot be quickly undone or that touches regulated data at scale. Compliance records become a natural output of the architecture, not a parallel documentation effort.

7.2. Expected Operational Outcomes

A Zero Trust operations model is credible only if it produces measurable improvements in outcomes rather than merely increasing control count. This paper presents a framework and its formal structure, not deployment or simulation results; we therefore state the anticipated effects as four explicit, falsifiable hypotheses (H1-H4), each with its evaluation metric and baseline—throughout, the organization's pre-MST access-control regime, typically static role-based access with standing privilege, measured on the same metrics over a comparable window. Section 7.7 identifies the corresponding validation designs.

H1: Reduction in unjustified access. Progressive removal of standing trust, JIT privilege, and device-trust tiering shrink the attack surface available to an adversary holding a valid credential, directly reducing the dwell time and lateral movement a credential-theft event implies; credential and identity-based attacks remain the dominant initial-access vector [10]-[12]. *Metric*: the measured trust surplus—the count and risk-weighted sum of permitted actions outside the reconciled legitimate set—plus standing-privilege counts and the lateral reach of a simulated compromised credential in red-team exercises, before and after MST adoption and against the static-RBAC baseline.

H2: Reduction in analyst toil. Unified telemetry and bounded AI automation reduce the manual correlation work imposed by fragmented detection stacks; alert

enrichment, case summarization, and duplicate suppression are the highest-volume, lowest-value analyst tasks, and automating them frees capacity for threat hunting, policy refinement, and incident command. *Metric*: percentage of analyst time on repeatable low-value tasks, alerts handled per analyst-hour, and the fraction of alerts arriving with sufficient context to support a decision without manual pivoting, in before-and-after or shadow-mode comparison.

H3: Improvement in detection and containment speed. A shared telemetry pipeline with pre-enriched incident records eliminates the context-assembly phase between alert and containment decision. *Metric*: mean time to detect (MTTD) and mean time to contain (MTTC), measured against the same incident classes under the pre-MST alerting pipeline, with false-block rate tracked simultaneously so that speed gains cannot be purchased invisibly with precision losses.

H4: Improvement in audit-evidence generation. Unified telemetry with compliance-tagged event records turns audit preparation from retrospective evidence-gathering into a continuous operational byproduct: access logs, control outcomes, exception records, and policy decisions exist as natural platform outputs. *Metric*: evidence latency (time to reconstruct a specific access event for audit), control-to-case traceability coverage, and preparation hours per audit cycle, against the retrospective baseline.

7.3. Performance Metrics and Governance Cadence

The appropriate scorecard for MST balances security effectiveness, analyst efficiency, business friction, and evidence quality. Useful quantitative measures include mean time to detect, contain, and recover (MTTD, MTTC, MTTR); false-block rate (automated blocks later reversed as erroneous); exception-turnaround time; percentage of analyst time on repeatable low-value tasks; percentage of autonomous actions subsequently reversed; and the fraction of alert volume arriving with sufficient context to support a decision without manual pivoting.

In regulated environments, a further class of metrics is operationally essential: evidence latency (how quickly can an event be reconstructed for audit?); control-to-case traceability (can every control action be linked to a governing policy and a ticket?); and answer time for auditor or customer inquiries about specific access events.

Calibrating the model parameters. The reviewable question "where do the numbers come from?" deserves a procedural answer: (i) assign c_i from the organization's existing data classification on a relative tier scale such as {1, 10, 50, 100}—the ratios, not the absolute values, drive the optimization; (ii) elicit d_i from the business owners of affected workflows as the relative cost of a blocked action, anchored to concrete scenarios; (iii) set $\theta(\rho_i)$ per resource sensitivity class as in Section 6.2; (iv) sweep the global permissiveness family of **Figure 5** on historical access logs to sensitivity-check the operating point before enforcement; and (v) re-run the sweep at governance cadence, because p_i, $\mathcal{L}$, and the cost structure drift. Steps (i)-(iii) reuse artifacts regulated enterprises already maintain, which

makes the calibration practicable; the sweep in step (iv) makes the policy's dependence on calibration visible and reviewable rather than implicit.

An explicit multi-objective optimization model is useful for framing governance reviews honestly. Security teams are often informally evaluated on whether they block enough adversarial activity while formally measured on availability, audit readiness, and customer trust. These objectives can pull in opposite directions, and the temptation is to resolve the tension implicitly rather than explicitly. The MST optimization structure (Section 6.3, Equation (4)) makes the tradeoff visible: reducing $|\mathcal{T} \setminus \mathcal{L}|$ is the security objective; maintaining $\mathcal{L} \subseteq \mathcal{T}$ is the availability constraint; and the verification threshold θ parameterizes the risk tolerance of the organization at each resource class. Governance reviews should track all three dimensions simultaneously.

7.4. Validation and Red-Team Considerations

Validation must be explicit and structured. Edge policies should be exercised through red-team and purple-team scenarios simulating exploit traffic, abuse-list hits, false-attribution cases, and partner-traffic exceptions; internal controls tested against stolen-session scenarios, unmanaged-device attempts, privilege-escalation paths, support-account misuse, and insider-style anomalies; and AI-enabled playbooks run in shadow mode—with before-and-after comparison of analyst actions, response speed, reversal rates, and business impact—before promotion to live execution authority.

Without structured validation, organizations tend to overestimate both detection precision and readiness for autonomous action. The bounded-autonomy ladder in **Figure 4** and the decision matrix in **Table 1** are useful not just as design guides but as evaluation rubrics: an organization can assess whether its current autonomous action program stays within the cells that the framework designates as safe for automation.

These reviews should engage more than the SOC: in a regulated SaaS company the consequences of Zero Trust policy are distributed across engineering, support, privacy, compliance, product, and customer trust, so a regular cross-functional governance cadence is essential. At minimum, review the precision of automated blocks, the exception patterns introduced by geo or reputation controls, the policy drift from repeated emergency overrides, the failure modes of behavioral detections, and the cases where bounded automation prevented harm or created avoidable friction—converting isolated lessons into structured policy refinement.

7.5. Limitations and Scope Boundaries

The MST framework addresses operational architecture and governance structure; it does not prescribe tool vendors, identity provider configurations, or compliance mappings, and organizations must adapt its principles to their regulatory context, technology stack, workforce scale, and risk tolerance. The formal model operates at two timescales, and its computability differs accordingly: the per-action thresh-

old rule (Equation (9)) is designed for real-time enforcement in constant time per decision, while the capacity-coupled planning problem (Equation (10)) is solved offline at governance cadence with standard MIP tooling. What the paper does not supply is empirical validation on deployment or simulation data: $V(a)$, θ, c_i, and d_i are policy parameters each organization must calibrate, and the outcome hypotheses (Section 7.2) remain to be tested against an access-control baseline—identified in Section 7.7 as the priority research direction.

The framework also does not resolve the challenge of third-party and supply-chain trust. SaaS providers depend on cloud infrastructure, open-source components, and vendor integrations that introduce trust relationships outside the direct control of the enterprise security program. While the two-plane architecture provides a structural basis for evaluating third-party access, a full supply-chain Zero Trust model is a substantial research and operational program in its own right, and is identified as a direction for future work.

7.6. Synthesis of the Theoretical Framework

The formal development in Section 6 can be summarized in five interlocking elements that together transform Zero Trust from a posture into an explicit constrained-optimization program.

The objective function. Equation (4) expresses MST as the minimization of a two-term cost—risk exposure from permitted actions outside the business-required set plus operational friction from denied legitimate actions—subject to completeness, verification sufficiency, and integrality. Under complete information this specifies the target policy in closed form; under legitimacy uncertainty the expected-cost form (Equation (8)) makes the two terms trade off at every action and yields the per-decision rule of Equation (9). The weights c_i, d_i and thresholds $\theta(\rho_i, t)$ are policy parameters organizations calibrate; the framework specifies their roles, not their values.

The decision variables. The action tuple $a_i = (\iota_i, \delta_i, \rho_i, \sigma_i, \omega_i)$ captures the five dimensions an MST-aligned policy must consider for every access decision: identity, device tier, resource and data classification, session context, and requested operation. Treating these jointly—rather than as independent gates—is what distinguishes MST from earlier RBAC, ABAC, and policy-administrator-only formalizations.

The graph formulation. The action-space-time graph $G^{(t)} = (V_G, E_G^{(t)})$ casts the MST optimum as an edge-selection problem over business-required edges under time-varying verification weights, making time explicit, connecting to provenance-graph datasets, and providing the substrate for the uncertainty-weighted and capacity-coupled variants.

The complexity separation. The complete-information problem is solved in closed form (Proposition 1); the expected-cost problem is solved exactly by a constant-time threshold rule (Equation (9)); and the capacity-coupled planning problem is NP-hard by exact reduction from 0-1 knapsack (Proposition 2), with the

pseudo-polynomial dynamic program transferring directly and the FPTAS and greedy guarantees applying to the complementary avoided-denial objective. Cheap enforcement plus hard-but-offline planning is the division of labour that makes the model deployable.

The attack-surface implication. The MST permitted action set $\mathcal{T}^*$ *is* the enterprise attack surface in the established Manadhata-Wing sense. The MST objective is therefore a direct quantitative expression of attack surface minimization under operational feasibility constraints, with reputation-based pre-filtering shrinking the effective action space $\mathcal{S}_{\text{eff}}$ before optimization. Every gap between the realized $\mathcal{T}^*$ and the business-required minimum $\mathcal{L}$ is a measurable architectural deficit rather than an unmeasurable policy judgement.

7.7. Future Work

Six research directions emerge directly from the framework.

A controlled synthetic simulation study. The nearest-term extension roots the model in the outcome-based framework of Section 7.2 before any production deployment: generate synthetic action populations with heterogeneous (p_i, c_i, d_i, w_i) drawn from the calibration tiers of Section 7.3, including controlled miscalibration of p_i and capacity stress on B; execute three policies—the MST rule of Equation (9) with the planner of Equation (10), a static-RBAC baseline with standing privilege, and a risk-adaptive policy without the MST cost semantics—and score each on the H1-H4 metrics. Every quantity is an object the model already defines, so the study isolates the marginal contribution of the cost semantics and the verification gate and yields sensitivity surfaces that inform the parameter-governance procedure of Section 7.3; a clean result is the natural gate before the field study described next.

Empirical validation of the proxy hypothesis. The foundational modelling assumption of MST—that minimising the risk-weighted trust surplus is a reliable proxy for reducing breach probability or expected breach impact—is operationally motivated but not proven. Longitudinal study of Zero Trust maturity against observed security outcomes is the primary research direction; the growing corpus of enterprise ZT deployment data identified in the multivocal review [16] makes such study increasingly tractable.

Provenance-graph instantiation of the action-space-time graph. The MST edge-selection formulation can be instantiated on public provenance-graph datasets (e.g., those used in OmegaLog [46]), treating each system-call-level event as a candidate x_i and applying the verification and risk constraints per edge. This would provide the first quantitative comparison of MST-based access policy against baseline access-control regimes—static RBAC with standing privilege, and risk-adaptive authorization without the MST cost semantics—at realistic granularity, directly testing hypotheses H1 and H3 of Section 7.2.

Action-independence extensions. The current formulation treats each action as independent. Extending the model to capture combinatorial interactions—for in-

stance, with pairwise risk terms $c_{ij}x_i x_j$ for action pairs known to compose into higher-risk compound actions (a read on regulated data plus an export to an external endpoint, for example)—substantially increases computational complexity but is essential for capturing real exfiltration paths. A principled treatment of compound-action risk is an important next step.

Learning the legitimate set $\mathcal{L}$ from operational behaviour. Organizations maintain $\mathcal{L}$ through policy declarations that lag operational reality. Time-indexed Bayesian updates, semi-supervised clustering, or other behavioural-learning approaches could close the gap between declared and effective legitimate sets, and would integrate naturally with the UEBA evidence layer [23] that already feeds the verification score $V(a_i, t)$.

Supply-chain extension. The current two-plane architecture addresses the enterprise's direct attack surface. SaaS providers depend on cloud infrastructure, open-source components, and vendor integrations that introduce trust relationships outside direct control. Extending MST to fourth-party trust—where actions a_i may be initiated by entities whose identity and device tier are not directly observable to the enterprise—is a substantial research program in its own right and a natural next direction.

8. Conclusions

Zero Trust is not a product category. Network location is not a trust signal. Standing privilege is the enemy. Compliance evidence is a byproduct of good architecture, not its purpose.

These four commitments are not rhetorical; each has an operational consequence. Treating Zero Trust as the progressive elimination of standing trust—not a product category—forces every grant to be justified, evidenced, and time-bounded. Treating unified telemetry as the precondition for trustworthy automation and audit-quality evidence forces schema and provenance discipline at ingestion. Treating bounded autonomy as a policy artifact with explicit rungs forces every autonomous action to carry a reversibility class and approval tier. And treating compliance as an output of operations forces control outcomes to be recorded as they occur.

In a regulated SaaS enterprise, the Minimum Standing Trust framework operationalizes these commitments: deny-by-default at the exposed application edge; identity-, device-, and session-aware internal control with JIT privilege throughout; unified telemetry across detection and access systems; and policy-governed AI autonomous response bounded by the same trust criteria that govern human access.

The MST framework presented in this paper makes four contributions to the Zero Trust literature.

First, it reframes the Zero Trust implementation problem as a constrained optimization with competing objectives—minimizing unnecessary trust while preserving operational completeness—and shows why monotonic control accumula-

tion fails as a strategy. This reframing explains empirically documented implementation failures that prior frameworks left without a structural account [16].

Second, it prescribes asymmetric enforcement posture across the two planes and sharpens the control hierarchy: WAF, NIDS, and reputation controls are edge instruments, not strategy; identity, device posture, JIT privilege, and session verification are the substance of Zero Trust in the internal enterprise.

Third, it establishes unified telemetry as a correctness requirement of the MST model—not a convenience—and provides a four-stage pipeline architecture (Decoder, Alerting, Enrichment, Enriched Incident) that operationalizes this requirement at the decision layer rather than at the dashboard layer.

Fourth, it integrates AI governance into the MST model as a first-class concern, showing that the same trust criteria governing human access decisions govern AI autonomous response actions, and providing the bounded-autonomy ladder as a practical policy instrument for both.

A consequence of these four contributions, developed formally in Sections 6.1-6.6, is that the MST optimum corresponds to a principled minimum of the enterprise attack surface in the established attack-surface sense [27]: the permitted action set is the smallest attacker-exploitable surface consistent with business requirements, and bidirectional reputation-based blocking—applied symmetrically at perimeter and endpoint—further shrinks the effective action space by removing actions whose counterparty is already known to be adversarial.

This operating model does not solve every problem, but it closes the gap between Zero Trust intent and Zero Trust operations: the edge is governed by explicit deny-by-default policy, the enterprise by continuously re-verified minimal trust, the telemetry plane by one shared evidentiary record, and automation by an explicit, reviewable autonomy boundary—with compliance evidence generated as a byproduct rather than reconstructed after the fact.

For organizations operating under the mobile, distributed, and data-regulated conditions that now characterize modern enterprise computing, the MST framework offers a practical and theoretically grounded path to the intersection of Zero Trust principles and effective cyber defense operations.

Future work. The most pressing empirical question is whether the MST model's core hypothesis holds in practice: that eliminating standing trust—measured by credential exposure, dwell time, and lateral movement reach—reliably proxies reduced breach impact. Longitudinal studies building on the multivocal review methodology of Buck *et al.* [16] would provide a rigorous path to this validation; instantiating the action-space-time graph on provenance-graph datasets [46] would supply a concrete experimental substrate for the MST edge-selection formulation at system-call granularity; efficient MIP formulations calibrated to enterprise scale are a tractable near-term algorithmic agenda; and the extension to supply-chain trust and the governance of agentic AI—irreversibility constraints, liability for autonomous access decisions, regulatory standards—remain open questions at the intersection of computer science and law.

Data Availability

No datasets were generated or analyzed during the preparation of this conceptual and operational framework paper.

Acknowledgements

The authors thank colleagues at Florida Atlantic University's Department of Electrical Engineering and Computer Science and at Athena Security Group for discussions that informed this work.

Conflicts of Interest

All three authors are affiliated with Athena Security Group. Peter J. Worth Jr. is founder, president, and CEO of Athena Security Group, and is concurrently a PhD candidate in the Department of Electrical Engineering and Computer Science at Florida Atlantic University. Dr. Ionut Cardei is a Professor in the Department of Electrical Engineering and Computer Science at Florida Atlantic University and serves as Chief Scientist at Athena Security Group. Dr. Chase Cunningham serves on the Leadership team of Athena Security Group and is the founder of the Dr. Zero Trust podcast. Athena Core and Athena Pallas are discussed in the paper only as disclosed implementation examples of the unified telemetry and AI-assisted response patterns described by the Minimum Standing Trust framework; the framework is intended to be vendor-neutral and applicable beyond any specific product or platform.

References

[1] Rose, S., Borchert, O., Mitchell, S. and Connelly, S. (2020) Zero Trust Architecture. NIST Special Publication 800-207. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-207

[2] Kindervag, J. (2010) Build Security into Your Network's DNA: The Zero Trust Network Architecture. Technical Report, Forrester Research.

[3] Ward, R. and Beyer, B. (2014) BeyondCorp: A New Approach to Enterprise Security. *USENIX ;login:*, **39**, 6-11.

[4] Osborn, B., McWilliams, J., Beyer, B. and Saltonstall, M. (2016) BeyondCorp: Design to Deployment at Google. *USENIX ;login:*, **41**, 28-34.

[5] Gilman, E. and Barth, D. (2017) Zero Trust Networks: Building Secure Systems in Untrusted Networks. O'Reilly Media.

[6] Chandramouli, R. (2023) A Zero Trust Architecture Model for Access Control in Cloud-Native Applications in Multi-Cloud Environments. NIST Special Publication 800-207A, National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-207A

[7] Cybersecurity and Infrastructure Security Agency (2023) Zero Trust Maturity Model, Version 2.0. Technical Report, CISA.

[8] United States Department of Defense (2022) Department of Defense Zero Trust Strategy. Technical Report, Office of the Chief Information Officer, U.S. Department of Defense.

[9] Executive Office of the President (2021) Executive Order 14028: Improving the Nation's Cybersecurity. *Federal Register*, **86**, 26633.

[10] Verizon (2025) 2025 Data Breach Investigations Report. Technical Report, Verizon Communications, 18th Edition. https://www.verizon.com/business/resources/reports/dbir/

[11] CrowdStrike (2025) 2025 CrowdStrike Global Threat Report. Technical Report, CrowdStrike. https://www.crowdstrike.com/global-threat-report/

[12] Cisco Talos Intelligence Group (2025) Cisco Talos Year in Review 2024. Technical Report, Cisco Systems. https://blog.talosintelligence.com/

[13] Strom, B.E., Applebaum, A., Miller, D.P., Nickels, K.C., Pennington, A.G. and Thomas, C.B. (2018) MITRE ATT&CK: Design and Philosophy. Technical Report MP180360, The MITRE Corporation.

[14] Saltzer, J.H. and Schroeder, M.D. (1975) The Protection of Information in Computer Systems. *Proceedings of the IEEE*, **63**, 1278-1308. https://doi.org/10.1109/proc.1975.9939

[15] Cunningham, C. (2018) The Forrester Wave: Zero Trust eXtended (ZTX) Ecosystem Providers, Q4 2018. Technical Report, Forrester Research.

[16] Buck, C., Olenberger, C., Schweizer, A., Völter, F. and Eymann, T. (2021) Never Trust, Always Verify: A Multivocal Literature Review on Current Knowledge and Research Gaps of Zero-Trust. *Computers & Security*, **110**, Article ID: 102436. https://doi.org/10.1016/j.cose.2021.102436

[17] Beyer, B., Beske, C.M., Peck, J. and Saltonstall, M. (2017) Migrating to BeyondCorp: Maintaining Productivity While Improving Security. *USENIX ;login:*, **42**, 49-55.

[18] Syed, N.F., Shah, S.W., Shaghaghi, A., Anwar, A., Baig, Z. and Doss, R. (2022) Zero Trust Architecture (ZTA): A Comprehensive Survey. *IEEE Access*, **10**, 57143-57179. https://doi.org/10.1109/access.2022.3174679

[19] de Chaves, S.A., Westphall, C.B. and Lamin, F.R. (2010) SLA Perspective in Security Management for Cloud Computing. 2010 *Sixth International Conference on Networking and Services*, Cancun, 7-13 March 2010, 212-217. https://doi.org/10.1109/icns.2010.36

[20] Ratchford, M., El-Gayar, O., Noteboom, C. and Wang, Y. (2022) BYOD Security Issues: A Systematic Literature Review. *Information Security Journal: A Global Perspective*, **31**, 253-273. https://doi.org/10.1080/19393555.2021.1923873

[21] Sandhu, R.S. and Samarati, P. (1994) Access Control: Principle and Practice. *IEEE Communications Magazine*, **32**, 40-48. https://doi.org/10.1109/35.312842

[22] Hu, V.C., Ferraiolo, D., Kuhn, R., Schnitzer, A., Sandlin, K., Miller, R. and Scarfone, K. (2014) Guide to Attribute Based Access Control (ABAC) Definition and Considerations. NIST Special Publication 800-162, National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-162

[23] Tuor, A., Kaplan, S., Hutchinson, B., Nichols, N. and Robinson, S. (2017) Deep Learning for Unsupervised Insider Threat Detection in Structured Cybersecurity Data Streams. *Proceedings of the AAAI-17 Workshop on AI for Cyber Security*, San Francisco, 4-5 February 2017, 224-231.

[24] Liu, A.X. and Gouda, M.G. (2008) Diverse Firewall Design. *IEEE Transactions on Parallel and Distributed Systems*, **19**, 1237-1251. https://doi.org/10.1109/tpds.2007.70802

[25] Shandilya, V., Simmons, C.B. and Shiva, S. (2014) Use of Attack Graphs in Security

Systems. *Journal of Computer Networks and Communications*, **2014**, Article ID: 818957. https://doi.org/10.1155/2014/818957

[26] Howard, M., Pincus, J. and Wing, J.M. (2005) Measuring Relative Attack Surfaces. In: Lee, D.T., Shieh, S.P. and Tygar, J.D., Eds., *Computer Security in the* 21*st Century*, Springer-Verlag, 109-137. https://doi.org/10.1007/0-387-24006-3_8

[27] Manadhata, P.K. and Wing, J.M. (2011) An Attack Surface Metric. *IEEE Transactions on Software Engineering*, **37**, 371-386. https://doi.org/10.1109/tse.2010.60

[28] AbuseIPDB (2024) Community-Driven IP Address Abuse Reporting and Reputation Database. Marathon Studios Inc. https://www.abuseipdb.com/

[29] The Spamhaus Project (2024) Don't Route or Peer (DROP) and Extended DROP (EDROP) Lists. The Spamhaus Project. https://www.spamhaus.org/drop/

[30] Husari, G., Al-Shaer, E., Ahmed, M., Chu, B. and Niu, X. (2017) TTPDrill: Automatic and Accurate Extraction of Threat Actions from Unstructured Text of CTI Sources. *Proceedings of the* 33*rd Annual Computer Security Applications Conference*, Orlando, 4-8 December 2017, 103-115. https://doi.org/10.1145/3134600.3134646

[31] Ferrag, M.A., Alwahedi, F., Battah, A., Cherif, B., Mechri, A., Tihanyi, N., *et al.* (2025) Generative AI in Cybersecurity: A Comprehensive Review of LLM Applications and Vulnerabilities. *Internet of Things and Cyber-Physical Systems*, **5**, 1-46. https://doi.org/10.1016/j.iotcps.2025.01.001

[32] Motlagh, F.N., Hajizadeh, M., Majd, M., Najafi, P., Cheng, F. and Meinel, C. (2024) Large Language Models in Cybersecurity: State-of-the-Art. arXiv: 2402.00891. https://doi.org/10.48550/arXiv.2402.00891

[33] Tabassi, E. (2023) Artificial Intelligence Risk Management Framework (AI RMF 1.0). NIST AI 100-1, National Institute of Standards and Technology.

[34] Autio, C., Schwartz, R., Dunietz, J., Jain, S., Stanley, M., Tabassi, E., Hall, P. and Roberts, K. (2024) Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile. NIST AI 600-1, National Institute of Standards and Technology. https://doi.org/10.6028/NIST.AI.600-1

[35] National Security Agency, Cybersecurity and Infrastructure Security Agency, Federal Bureau of Investigation, Australian Cyber Security Centre, New Zealand National Cyber Security Centre and United Kingdom National Cyber Security Centre (2024) Deploying AI Systems Securely: Best Practices for Deploying Secure and Resilient AI Systems. Technical Report, Joint Cybersecurity Information.

[36] National Security Agency, Cybersecurity and Infrastructure Security Agency, Federal Bureau of Investigation, Australian Signals Directorate's Australian Cyber Security Centre and International Partners (2025) AI Data Security: Best Practices for Securing Data Used to Train and Operate AI Systems. Technical Report, Joint Cybersecurity Information.

[37] National Cyber Security Centre (2023) Guidelines for Secure AI System Development. Technical Report, NCSC.

[38] Scarfone, K. and Mell, P. (2007) Guide to Intrusion Detection and Prevention Systems (IDPS). NIST Special Publication 800-94, National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-94

[39] National Institute of Standards and Technology (2024) The NIST Cybersecurity Framework 2.0. Technical Report, National Institute of Standards and Technology. https://doi.org/10.6028/NIST.CSWP.29

[40] OWASP GenAI Security Project (2025) Securing Agentic Applications Guide 1.0.

Technical Report, OWASP Foundation.

[41] Weidinger, L., Uesato, J., Rauh, M., Griffin, C., Huang, P., Mellor, J., *et al.* (2022) Taxonomy of Risks Posed by Language Models. 2022 *ACM Conference on Fairness Accountability and Transparency*, Seoul, 21-24 June 2022, 214-229. https://doi.org/10.1145/3531146.3533088

[42] Abdelnabi, S., Greshake, K., Mishra, S., Endres, C., Holz, T. and Fritz, M. (2023) Not What You've Signed up for: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection. *Proceedings of the* 16*th ACM Workshop on Artificial Intelligence and Security*, Copenhagen, 30 November 2023, 79-90. https://doi.org/10.1145/3605764.3623985

[43] Papernot, N., McDaniel, P., Sinha, A. and Wellman, M. (2016) Towards the Science of Security and Privacy in Machine Learning. arXiv: 1611.03814.

[44] Parasuraman, R. and Riley, V. (1997) Humans and Automation: Use, Misuse, Disuse, Abuse. *Human Factors*: *The Journal of the Human Factors and Ergonomics Society*, **39**, 230-253. https://doi.org/10.1518/001872097778543886

[45] Lee, J.D. and See, K.A. (2004) Trust in Automation: Designing for Appropriate Reliance. *Human Factors*: *The Journal of the Human Factors and Ergonomics Society*, **46**, 50-80. https://doi.org/10.1518/hfes.46.1.50_30392

[46] Hassan, W.U., Noureddine, M.A., Datta, P. and Bates, A. (2020) OmegaLog: High-Fidelity Attack Investigation via Transparent Multi-Layer Log Analysis. *Proceedings* 2020 *Network and Distributed System Security Symposium*, San Diego, 23-26 February 2020, 1-18. https://doi.org/10.14722/ndss.2020.24270

[47] Garey, M.R. and Johnson, D.S. (1979) Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman.

[48] Kellerer, H., Pferschy, U. and Pisinger, D. (2004) Knapsack Problems. Springer. https://doi.org/10.1007/978-3-540-24777-7

[49] Wolsey, L.A. (1998) Integer Programming. Wiley-Interscience.

[50] McGregor, A. (2014) Graph Stream Algorithms: A Survey. *ACM SIGMOD Record*, **43**, 9-20. https://doi.org/10.1145/2627692.2627694

[51] Athena Security Group (2026) Athena Core. Athena Security Group.

[52] Athena Security Group (2026) Athena AI Analyst (Pallas). Athena Security Group.