Zero Trust: Decision Dependencies, Freshness, and Failure

CoreBond Research Series — CBRS-005
August 2026
Todd Kirton
DOI: 10.5281/zenodo.22029424

Abstract

Zero Trust is often summarized as “never trust, always verify,” but the architecture is more precise than the slogan. NIST frames Zero Trust around minimizing uncertainty, rejecting implicit trust based on network location or ownership, and making granular, least-privilege access decisions. Mature guidance already assumes that access can be reevaluated as identity, device posture, behavior, policy, and other information changes. The unresolved engineering question is what happens when the dependencies behind an evidence-driven decision are stale, unavailable, delayed, inconsistent, or compromised. This paper uses “decision dependency” as an analytical grouping for the identity, policy, telemetry, and enforcement systems whose state can materially affect an access outcome. It argues that Zero Trust’s shift from broad implicit inheritance to explicit, granular decision-making improves visibility and control while coupling access outcomes more directly to the quality of those dependencies. The resulting problem is one of explicit assumption management: how current must evidence be, what can safely persist when infrastructure is unavailable, how should distributed components bound disagreement, and what should force an existing decision to be reconsidered?

1. The Slogan Is Not the Architecture

“Never trust, always verify” works because it is memorable. Taken literally, however, it is a poor description of a system that must eventually decide whether an action is permitted. A system that grants access has concluded that the available evidence and policy are sufficient for some bounded purpose. The useful question is not whether that conclusion should be called trust. It is what the conclusion authorizes, how broadly it can be inherited, how long it remains acceptable, and what can cause it to change.

NIST SP 800-207 is deliberately narrower. Zero Trust is a resource-protection paradigm in which no implicit trust is granted merely because of physical or network location or because an asset is enterprise-owned. Its objective is to minimize uncertainty while enforcing accurate, least-privilege, per-request access decisions in an environment treated as potentially compromised (Rose et al., 2020). Zero Trust therefore does not promise perfect knowledge. It rejects broad shortcuts and asks the architecture to make more deliberate access decisions.

This paper begins after that point. NIST and later implementation guidance already recognize that access decisions consume changing information. The question pursued here is what happens when the dependencies behind that decision loop fail in different ways. If access is meant to respond to current identity, posture, policy, and security conditions, then the age, integrity, availability, and consistency of those inputs become part of the decision itself.

2. What Zero Trust Actually Removes

Traditional enterprise security often allowed location or network membership to carry substantial authority. A user or device that crossed the expected boundary could inherit confidence from being "inside." Zero Trust rejects that inheritance as sufficient grounds for access. Location and ownership can remain policy inputs, but neither is supposed to substitute for evaluating the subject, device, requested resource, and applicable conditions (Rose et al., 2020).

The result is a move toward resource-specific authority. NIST describes access as granular and per-session, with authentication and authorization performed before a session to an enterprise resource is established. Permission to reach one resource does not automatically grant permission to another. Dynamic policy can consider identity, asset state, resource characteristics, environmental information, and behavior. The architectural improvement is not that assumptions vanish; it is that fewer assumptions are allowed to silently confer broad authority.

That matters because inherited authority can outlive the evidence that originally justified it. Zero Trust repeatedly brings the question back into view: this subject, from this device, under these conditions, requesting this resource. The narrower the answer, the less damage one stale or compromised assumption can automatically propagate.

3. A Decision Still Has to Be Made

Zero Trust does not collapse authentication and authorization into one act. Authentication establishes a subject or device under the applicable mechanism. Authorization determines whether the authenticated subject may perform a specific action against a specific resource. NIST treats those as distinct functions and places the access decision inside a broader policy process rather than treating successful authentication as sufficient permission (Rose et al., 2020).

In NIST's logical architecture, a Policy Engine reaches the decision, a Policy Administrator carries that decision into the access path, and a Policy Enforcement Point enables, monitors, or terminates the connection. An approved session can later be limited or revoked. NIST's implementation work extends this idea into ongoing access: information about identity, endpoint hygiene, behavior, and other conditions can continue to inform whether access should persist (NIST NCCoE, 2025).

Zero Trust is therefore best understood here as a decision architecture operating under uncertainty. Its value depends not only on whether a credential was valid at some earlier instant but on whether the overall access decision remains appropriate for the resource and action. That makes the systems supplying and enforcing the decision worth examining in their own right.

4. Continuous Verification Is a Decision Loop

Continuous verification is sometimes flattened into the image of a user or device repeatedly logging in. That is too narrow. Zero Trust guidance describes an ongoing decision process in which access can respond to changes in identity information, device condition, behavior, resource state, policy, and other security information. CISA's maturity model similarly treats visibility, analytics, automation, and governance as capabilities that cut across Zero Trust pillars rather than reducing the model to repeated credential checks (CISA, 2023).

NIST SP 1800-35 makes the ongoing nature of the process especially useful for this paper. Its implementation architecture describes continuing collection of relevant information during access and

continued policy decisions about whether that access should remain permitted, be limited, or be revoked (NIST NCCoE, 2025). The important object is therefore not the login ceremony. It is the continuing validity of the access decision.

Current NSA device guidance reinforces the same point from the device side. The Device Pillar treats devices as subjects of continuing authentication, inspection, assessment, and posture evaluation, including compromise status, software state, protections, encryption settings, and configuration integrity (NSA, 2026). Mature Zero Trust guidance is already well beyond a one-time device check.

The useful model is a loop: evidence is obtained or referenced, policy evaluates it for a requested action, enforcement applies the result, and later information can alter the result. The timing and granularity vary by implementation. What does not vary is the underlying dependency on information and control state that are assumed to be good enough for the decision being made.

5. The Dependencies Behind the Decision

For this paper, a decision dependency is an analytical grouping, not a new Zero Trust component. It means any identity, policy, telemetry, enforcement, or supporting state whose integrity, currency, or availability can materially affect an access outcome. NIST already gives these systems their own architectural roles. Grouping them as decision dependencies simply allows their failure properties to be compared without pretending that an identity provider, a Policy Information Point, and a Policy Enforcement Point perform the same function.

The distinction matters because making a decision explicit does not make it independent. Identity information must be obtained. Device posture must be measured or reported. Policy must exist in an applicable form. Threat information, analytics, resource classification, or environmental context may influence the outcome. Enforcement must then make the policy result real.

None of this is a defect unique to Zero Trust. Perimeter architectures also depended on directories, VPNs, firewalls, endpoint controls, certificate infrastructure, routing, and administrative policy. Zero Trust can improve the security argument by narrowing authority, increasing observability, and permitting access to change as conditions change. What it also does is couple granular access outcomes more visibly to the quality of the decision process.

A policy engine can reason correctly from bad identity state. A legitimate posture report can be too old to describe the device now. A revocation can exist in one system while an enforcement point has not yet received it. The architecture can be functioning exactly as implemented while the decision is still based on information that is incomplete, old, delayed, or inconsistent. That is the problem this paper is interested in.

6. Freshness Is an Engineering Question, Not a Universal Timer

Zero Trust's continual-evaluation model creates a freshness question, but the standards do not provide one universal expiration interval for security evidence. That absence is sensible. The useful lifetime of information depends on what the information represents, the protected resource, the consequence of a wrong decision, the cost of obtaining newer state, and the operating environment.

A device posture result can be authentic and still cease to describe the device after compromise. Identity state can change after a session begins. Policy can be revised. Threat information can change. These examples do not establish a universal Zero Trust freshness rule; they show why an architecture that responds to changing conditions must decide when prior information is no longer sufficient.

This paper uses a working failure taxonomy rather than claiming an established or exhaustive one. Evidence may be stale: once valid, but no longer current enough. It may be unavailable: unreachable when the decision needs it. It may be delayed: current at origin but arriving too late to influence the relevant decision. Distributed components may be inconsistent. A source or control component may be compromised. These are different problems and can justify different responses.

A separate class concerns decision quality rather than evidence condition: an access decision can be over-scoped even when every input is current and legitimate. That distinction matters. Fresh evidence does not guarantee least privilege, just as perfect policy cannot compensate for corrupted evidence.

7. Availability Forces Assumptions to Persist Somewhere

Demanding newer information can improve currentness while increasing dependence on reachable services. That creates a design trade rather than a universal rule. If an implementation requires a fresh remote signal for every sensitive action, loss of that signal can become loss of access. If an implementation instead permits previously established state to remain usable for some bounded period, it preserves availability by accepting persistence.

This does not assume that Zero Trust standards prescribe caching or offline authorization. It follows from the design choice itself: if local or retained state temporarily substitutes for unavailable current state, the system has chosen an availability boundary and an age boundary at the same time. The security question is whether that trade is appropriate for the resource and consequence involved.

This is why “fail open” and “fail closed” are inadequate as universal slogans. Denial may be the safest outcome for one resource and the dangerous outcome for another. A payroll portal, an emergency communications function, and an industrial safety process do not share the same consequence model. The architecture has to know which assumptions may persist during degraded operation and which decisions require current evidence regardless of availability cost.

8. Distribution Helps Only When Failure Domains Are Actually Different

Another easy criticism imagines a single giant Policy Engine becoming the new perimeter. NIST’s implementation work does not require that topology. Policy decision and administration functions can be distributed across identity, endpoint, network, data-security, and other products, with enforcement occurring near different resources and workloads (NIST NCCoE, 2025).

Distribution can reduce some concentration risks when the components are genuinely separated across failure domains. It can support local enforcement, redundancy, resource-specific controls, and narrower blast radii. But distribution is not automatically resilience. Five policy components that all depend on the same identity provider, certificate authority, cloud control plane, or administrative authority can still share a common-mode failure.

Distribution also creates a synchronization problem. Different enforcement points can hold different policy versions. One system can learn of a device compromise before another receives the information. Identity can be revoked centrally while a local component continues from older state. Distributed systems will sometimes disagree; the security question is how that disagreement is bounded.

Which resources can proceed when state diverges? How long can two policy views remain different? Which change must propagate immediately? What happens when the system cannot determine which copy is current? Those are ordinary distributed-systems questions with direct authorization consequences.

9. The Device Hypothesis Did Not Survive

This research began with a suspicion that Zero Trust might stop too early at the device: authenticate the device, establish that it belongs, and then focus continuous evaluation elsewhere. The evidence defeated that broad hypothesis. Current NSA guidance treats device authentication, inspection, assessment, security posture, isolation, remediation, and control as ongoing concerns associated with access (NSA, 2026).

That result matters methodologically. When the evidence kills the planned critique, the critique should narrow rather than move the goalposts. Device posture is already in the Zero Trust loop.

A narrower distinction between the continuing evaluation of device posture and the basis by which a system recognizes device identity remains worth studying, but Paper 005 does not answer it. It is left as a separate research boundary rather than used here to manufacture a gap.

10. OT Shows Why Consequence Changes the Answer

Enterprise Zero Trust implementation guidance cannot simply be treated as a universal deployment recipe. NIST's SP 1800-35 implementation project excludes industrial control systems, operational technology, and IoT from its scope, while NIST SP 800-82 emphasizes that OT cybersecurity must account for performance, reliability, and safety requirements that differ from conventional enterprise IT (NIST, 2023; NIST NCCoE, 2025).

Consider a design scenario in which an access decision normally depends on a remote policy or posture service. In one environment, loss of that service may justify denying access until current state can be recovered. In a safety-critical operational environment, denial itself may create unacceptable consequence. The example does not prove a Zero Trust failure. It demonstrates why the consequence of unavailable evidence belongs in the access-policy design.

OT is therefore a boundary case for the freshness problem. It shows why one organization may demand extremely current evidence while another permits narrowly bounded local authority during degraded connectivity. The useful question is not whether Zero Trust works in OT in the abstract. It is how the architecture's principles are adapted when availability and safety are themselves security requirements.

11. What Zero Trust Actually Buys

Examining decision dependencies should not obscure the reason Zero Trust is useful. Broad implicit trust is operationally convenient until an attacker inherits it. If one compromised account, endpoint, or network position automatically opens large parts of an environment, the architecture has allowed yesterday's assumptions to carry too much authority.

Granular decisions can reduce that inheritance. Dynamic policy can account for current device condition and context. Enforcement can narrow access to a resource. New evidence can trigger limitation or revocation. Distributed controls can constrain some failure domains. Better visibility can expose assumptions that were previously buried in topology or membership.

Those benefits make the decision process more explicit and couple it more directly to granular access outcomes. The process always mattered. Zero Trust asks the organization to depend less on invisible inherited confidence and more on decisions it can observe, constrain, and revisit.

The operational obligation follows naturally: know which evidence materially affects which decisions, how current it needs to be, how disagreement is handled, how revocations propagate, and what happens when a dependency cannot be reached. Explicit dependencies are not automatically weaknesses. Unexamined dependencies are.

12. What Deserves to Be Discovered

The durable questions are now about evidence lifetime, provenance, disagreement, and recovery. How fresh must posture be for different classes of action? Which decisions can safely continue from retained state? How should a system express uncertainty when a normally required signal disappears? How should distributed enforcement reconcile conflicting policy views? What evidence should force an existing session to narrow or terminate?

There is also a recursion problem, but it does not require an infinite chain of verification. Identity systems, policy services, telemetry pipelines, enforcement components, roots of trust, administrators, and cryptographic anchors cannot all recursively verify one another forever. Every architecture eventually rests on assumptions.

The engineering goal is explicit assumption management: identify which roots and prior states must persist, limit the authority derived from them, monitor what can change, define what evidence invalidates a prior decision, and establish recovery when a dependency becomes suspect. The remaining assumptions should be visible enough to be challenged rather than silently inherited.

The more durable lesson behind Zero Trust is that yesterday's answer should not become today's unquestioned fact. Zero Trust does not reach a magical state in which nothing is trusted. It creates a discipline for asking whether the evidence supporting the current access decision is still good enough for the decision being made.

References

Cybersecurity and Infrastructure Security Agency. (2023). Zero Trust Maturity Model, Version 2.0. U.S. Department of Homeland Security.

National Institute of Standards and Technology. (2023). Guide to Operational Technology (OT) Security (NIST Special Publication 800-82 Rev. 3). U.S. Department of Commerce.

National Institute of Standards and Technology, National Cybersecurity Center of Excellence. (2025). Implementing a Zero Trust Architecture (NIST Special Publication 1800-35). U.S. Department of Commerce.

National Security Agency. (2026). Zero Trust Implementation Guidelines: Device Pillar. National Security Agency.

Rose, S., Borchert, O., Mitchell, S., & Connelly, S. (2020). Zero Trust Architecture (NIST Special Publication 800-207). National Institute of Standards and Technology.