International Journal of
Technology and
Systems
(IJTS)

Policy-as-Code and Compliance Automation for Secrets Governance in Agentic AI Workloads in Regulated Industries



Journal of Technology and Systems
ISSN: 2788-6344 (Online)
Vol. 8, Issue No. 4, pp 75 – 104, 2026
www.carijournals.org

Policy-as-Code and Compliance Automation for Secrets Governance in Agentic AI Workloads in Regulated Industries

Suhas Madupuri, Rajesh Thondamanati

Accepted: 20th Oct, 2026, Received in Revised Form: 10th Nov, 2026, Published: 20th Nov, 2026

Abstract

Purpose: This study examined the governance gap that opens when autonomous agents acquire, use, and propagate credentials dynamically, and proposed a reference framework, the Agentic Secrets Governance Framework (ASGF), to narrow it.

Methodology: The framework couples a SPIFFE and SPIRE based attestation layer with an Open Policy Agent decision point, a dynamic secrets broker that issues short lived, task scoped credentials, and a compliance control library that expresses obligations drawn from the HIPAA Security Rule, PCI DSS 4.0.1, the EU Digital Operational Resilience Act, the NIST AI Risk Management Framework, and ISO/IEC 42001 as machine evaluable Rego policies. A reference implementation was built and evaluated in a controlled three node Kubernetes testbed. Two workflows, healthcare eligibility determination and card payment fraud triage, were selected by purposive sampling, and ten thousand simulated requests were replayed per configuration.

Findings: Policy enforced, ephemeral credential issuance reduced the configured credential exposure window from months to minutes while adding a combined median of 34.3 milliseconds to the request path. In conformance testing, the policy layer denied the out of scope identity abuse and credential replay cases it was designed to deny, but, as expected, it did not stop goal hijack attacks that stayed within an agent's nominal scope.

Unique Contribution to Theory, Practice and Policy: The study frames secrets governance for agents as a joint security and compliance decision made at the moment of credential issuance. It gives practitioners a reusable control library pattern and issuance protocol, and it suggests that regulators could reduce interpretive risk by publishing machine evaluable reference controls alongside regulatory text. Residual risks from semantic intent verification and cross agent delegation are discussed.

Keywords: *Policy as Code, Secrets Management, Agentic AI, Non-Human Identity, Compliance Automation, Zero Trust, Regulated Industries*

Journal of Technology and Systems
ISSN: 2788-6344 (Online)
Vol. 8, Issue No. 4, pp 75 – 104, 2026
www.carijournals.org

1. Introduction

Autonomous, tool using artificial intelligence agents have moved from research demonstrations into production systems that authorize payments, adjudicate insurance claims, triage security incidents, and orchestrate infrastructure changes without a human confirming every step. Enterprise surveys conducted in early 2026 reported that eighty one percent of large organizations had fully adopted or were actively scaling agentic AI, and that all surveyed organizations planned further expansion within the year [1]. This shift changes what a credential is used for. A traditional service account authenticates a fixed, auditable piece of code that executes a narrow, previously reviewed function. An agentic workload, by contrast, decides at runtime which tool to call, which data to retrieve, and which downstream system to touch, and it may spawn sub agents that inherit or request credentials of their own. Peer reviewed studies have shown that such agents can be steered by instructions hidden in the content they retrieve [2], and benchmark studies have confirmed that tool using agents remain vulnerable to these attacks in realistic task settings [3], [4]. The identity, secrets, and policy infrastructure that regulated industries built over the past two decades, centered on long lived service accounts, manually reviewed access requests, and periodic compliance audits, was not designed for this pattern of dynamic, delegated, machine initiated authority.

The consequence is measurable. Large scale measurement research established several years ago that secret leakage in public repositories is pervasive, with thousands of new, unique secrets leaked every day [5]. The fifth annual State of Secrets Sprawl report found that 28.65 million hardcoded credentials were added to public GitHub repositories in 2025, a thirty four percent increase over the previous year, and that credentials associated with AI services grew eighty one percent over the same period, faster than any other credential category tracked in the report [6]. A companion industry whitepaper from the Cloud Security Alliance found that non human identities, which include the service accounts, API keys, and agent credentials that agentic systems depend on, outnumber human identities by a ratio of forty five to one across the average enterprise and by as much as one hundred forty four to one in cloud native environments, while only fifteen percent of surveyed organizations felt confident they could prevent an attack based on a compromised non human identity [7]. Fifty one percent of organizations reported no clear ownership for AI agent identities, and seventy eight percent lacked a documented policy governing when an agent identity should be created or removed [7]. These are not unrelated statistics. They describe the specific infrastructure gap that opens when agentic AI workloads are deployed onto identity and secrets management systems designed for static, human initiated access.

Regulated industries carry this gap differently from the rest of the economy. A healthcare payer operating under the HIPAA Security Rule must be able to show, for any access to protected health information, who or what accessed it, under what authorization, and for what purpose. A card issuer or payment processor operating under PCI DSS 4.0.1 must demonstrate that access to

cardholder data is authenticated, logged, and reviewed on a defined cycle. A bank or investment firm subject to the European Union's Digital Operational Resilience Act must be able to demonstrate that its information and communication technology risk management framework, including third party and AI related dependencies, is continuously tested and auditable. None of these frameworks was written with autonomous, tool calling AI agents in mind, and a recent structural analysis of AI identity standards found that current regulatory frameworks including SOC 2, ISO 27001, PCI DSS, and NIST 800-53 lack explicit provisions for agentic AI credentials, which allows an organization to pass a certification audit while leaving its agent identity governance almost entirely unaddressed [7], [8].

This paper addresses the gap by proposing a reference framework, which we call the Agentic Secrets Governance Framework (ASGF), that treats compliance obligations and secrets policy as executable code rather than as documentation to be reconciled with system behavior after the fact. ASGF combines three ideas that have each matured independently over the past several years: cryptographic workload identity through SPIFFE and SPIRE, declarative policy evaluation through the Open Policy Agent and its Rego language, and dynamic, short lived secrets issuance through a broker modeled on HashiCorp Vault's dynamic secrets engine and its recently introduced native agent authorization capabilities [9], [10]. The review of existing approaches in Section 6.3 shows that commercial platforms already provide many of these building blocks individually. The framework adds a fourth layer that the review did not find in any of the surveyed open source projects or product documentation: a compliance control library that expresses obligations drawn directly from the HIPAA Security Rule, PCI DSS 4.0.1, DORA, the NIST AI Risk Management Framework, and ISO/IEC 42001 as machine evaluable Rego policies, each linked to the regulatory provision it implements, so that every credential issuance decision is simultaneously a security decision and a compliance decision, evaluated once, logged once, and available as evidence for the next audit without manual reconciliation.

The contributions of this paper are as follows. First, we present a threat model for secrets governance in agentic AI workloads that extends the OWASP Top 10 for Agentic Applications with specific attention to credential handling failure modes. Second, we present the ASGF architecture and its runtime credential issuance protocol, including a mapping from framework controls to specific provisions in five regulatory and standards frameworks relevant to healthcare, payments, and financial services. Third, we describe a reference implementation and evaluate it in a controlled testbed modeled on two representative regulated industry workflows, a healthcare eligibility determination agent and a card payment authorization agent, measuring the latency overhead the framework adds to the request path and the reduction in credential exposure window compared to a static credential baseline. Fourth, we discuss the residual risks that policy as code and workload identity do not resolve, including the semantic intent verification problem identified in recent AI identity research, and we outline directions for standardizing agent identity and machine readable compliance evidence across organizational boundaries.

The remainder of the paper is organized as follows. Section 2 reviews related work in agentic AI security, non human identity management, and policy as code. Section 3 defines the problem and threat model. Section 4 presents the ASGF architecture. Section 5 describes the reference implementation and the two case studies. Section 6 presents the evaluation. Section 7 discusses implications and limitations. Section 8 outlines future work, and Section 9 concludes.

2. Background and Related Work

2.1 Agentic AI Systems and Their Threat Surface

An agentic AI system, for the purposes of this paper, is a software system in which a large language model or comparable planning component selects and invokes tools, retrieves and writes data, and sequences multi step tasks with limited or no human confirmation between steps. This definition follows the working description used by the Cloud Security Alliance's agentic AI governance research stream, which characterizes agents as actors that acquire permissions dynamically at runtime, spawn sub agents, invoke APIs, and chain actions across multiple systems, expanding the blast radius of any single compromised credential beyond what static credential holders present [7].

The academic literature has documented the attack surface of such systems in detail. Greshake et al. introduced indirect prompt injection, in which instructions planted in retrieved content take control of an LLM integrated application, and demonstrated the attack against production systems [2]. Zhan et al. developed InjecAgent to benchmark indirect prompt injection against tool integrated agents [3], Debenedetti et al. released AgentDojo, a dynamic environment of 97 realistic tasks and 629 security test cases for evaluating attacks and defenses [4], and Ruan et al. used a language model emulated sandbox to identify risky agent actions at scale [11]. A consistent finding across these studies is that the attack operates on the agent's reasoning, so it can succeed while every individual tool call remains formally valid. This finding defines the scope boundary of the present work.

The OWASP Foundation's Generative AI Security Project published the first Top 10 for Agentic Applications in December 2025, developed with more than one hundred industry contributors [12]. The list is organized around ten risk categories, denoted ASI01 through ASI10, spanning goal hijacking through hidden instructions, unsafe tool chaining, identity and privilege abuse across delegated authority, supply chain risk in dynamically loaded tools and models, unintended code execution, memory and context poisoning, insecure agent to agent communication, cascading failures across networked agents, exploitation of human trust in agent recommendations, and rogue agents that drift from their intended purpose while individual actions still appear legitimate [12]. Three of these ten categories, ASI03 (identity and privilege abuse), ASI04 (agentic supply chain vulnerabilities), and ASI07 (insecure inter agent communication), are directly about how an agent obtains, uses, and shares credentials, which is the specific problem this paper addresses.

Complementary threat modeling work has extended the MITRE ATLAS matrix, originally developed for adversarial machine learning, and introduced MAESTRO, a multi layer framework proposed by the Cloud Security Alliance for reasoning about multi agent system threats that separates concerns such as agent identity, inter agent communication, and the orchestration layer that coordinates multiple agents. A CSA gap analysis comparing MITRE ATT&CK and ATLAS against agentic control plane behavior found that a meaningful share of observed agent compromise techniques have no corresponding entry in either framework, because both were built around single model inference rather than multi step, credentialed, tool using behavior [13].

2.2 Secrets Management and Non-Human Identity

Secrets management as a discipline began with the recognition that hardcoded credentials in source code and configuration files are a persistent and preventable source of breaches. Meli et al. measured leakage across billions of GitHub files and showed that leaked secrets often remain recoverable from repository history even after removal [5], and Basak et al. systematically reviewed practitioner literature and identified 24 secret management practices, among which external secret management services and short lived secrets were the most recommended [14]. HashiCorp Vault popularized centralized secrets storage combined with dynamic secrets, in which the secrets engine generates a short lived, unique credential for each requesting client rather than distributing a shared long lived credential [9]. In June 2026, HashiCorp extended Vault with native support for AI agent authorization, requiring OAuth 2.0 Rich Authorization Requests by default [15], so that every authorization grant carries fine grained, per request scope rather than a broad, standing permission, and adding Terraform resources that let organizations declare agent identities and their ceiling policies as infrastructure as code [9]. Vault Enterprise also supports Sentinel policies tied to identities and request paths [16], and a SPIFFE authentication method [17]. HashiCorp has separately positioned the SPIFFE and SPIRE open specifications, now hosted by the Cloud Native Computing Foundation, as the identity layer agentic AI has been missing, because SPIFFE issues short lived, cryptographically verifiable workload identity documents (SVIDs) based on runtime attestation of the workload itself rather than a shared secret it happens to hold [10], [18]. This model implements the zero trust principle that no implicit trust is granted on the basis of network location or asset ownership [19]. An independent architecture paper on zero trust continuous integration and delivery systems reached a similar conclusion from the CI/CD domain, arguing that CI/CD runners have effectively become privileged automation agents that should replace static secrets with SPIFFE issued, attestation based identity integrated with policy engines such as Open Policy Agent and Cedar [20]; Cedar itself has been formally specified and verified, which shows that an authorization language can be both expressive and analyzable [21].

Work on agent identity and delegation has begun to address the same problem from the agent side. OAuth 2.0 Token Exchange defines how one party obtains a token to act on behalf of another

[22], and South et al. proposed extending such standards so that authority delegated from a human to an AI agent is authenticated, scoped, and auditable [23]. Syros et al. proposed SAGA, an implemented and formally verified architecture in which users register agents with a provider that enforces user defined access control policies on inter agent communication [24]. These proposals govern which agent may act and on whose behalf, but neither encodes external regulatory obligations into the authorization decision.

The scale of the non human identity problem has been quantified directly. The Cloud Security Alliance's May 2026 whitepaper on non human identity governance in agentic AI, drawing on a survey of enterprise security teams, found that only twenty percent of organizations have a formal offboarding process for API keys, that one in twenty non human identities carries full administrative privileges, and that between twenty four and thirty percent of organizations need more than twenty four hours to rotate an exposed credential [7]. The same report catalogued several 2026 incidents that illustrate the consequence of these gaps, including an OAuth token supply chain compromise that affected more than seven hundred customer environments and a credential harvesting incident that exposed 1.5 million agent tokens [7]. GitGuardian's 2026 secrets sprawl report adds a supply chain dimension, documenting the Shai-Hulud 2.0 npm worm, which Microsoft's security team separately analyzed as an aggressive, self propagating credential harvesting campaign that compromised software supply chain infrastructure at scale, with fifty nine percent of the compromised machines identified as CI/CD runners, precisely the automation identities the SPIFFE based CI/CD literature is concerned with [6], [25].

2.3 Policy as Code and Compliance Automation

Policy as code applies the same engineering discipline to authorization and compliance rules that infrastructure as code applied to system configuration: rules are expressed in a version controlled, testable, machine evaluable language rather than in a document that a human interprets. The Open Policy Agent (OPA) and its Rego policy language are the most widely adopted general purpose policy engine in the cloud native ecosystem, used to make authorization decisions for Kubernetes admission control, API gateways, and CI/CD pipelines by evaluating structured input against declarative rules and returning an allow or deny decision along with a machine readable explanation [26]. OPA based tools such as Gatekeeper and Conftest apply this model to configuration compliance at deployment time [27].

Applying policy as code specifically to regulatory compliance, rather than to general authorization, is a more recent and less thoroughly examined pattern. Kellogg et al. introduced continuous compliance, in which automated verification tools check compliance controls on every code change, and reported that auditors accepted the approach in an industrial setting [28]. A 2026 study of compliance as code for continuous PCI DSS 4.0 validation in cloud native financial systems categorized the pattern into four architectural approaches: translating regulatory provisions directly into policy as code rules, integrating those rules into the continuous integration

pipeline so that non compliant infrastructure changes are rejected before deployment, maintaining an immutable audit trail of every policy evaluation, and automating remediation of common violations [29]. The same study reported a production case in which combining an External Secrets Operator, pre deployment configuration validation, and Kubernetes admission control eliminated the window during which infrastructure could run in a non compliant state, and argued that this preventive enforcement pattern complements, rather than replaces, the detective monitoring and periodic audit activities that compliance teams already run [29]. This paper adopts the same four pattern taxonomy and extends it specifically to the credential and identity lifecycle of agentic AI workloads, which neither study addresses.

2.4 Regulatory Landscape for Regulated Industries

Three regulatory instruments most directly bound the case studies examined in Section 5. The HIPAA Security Rule requires covered entities and business associates handling protected health information to implement access controls, audit controls, and integrity controls proportionate to risk; the U.S. Department of Health and Human Services proposed a substantial update to the rule in January 2025 that would, among other changes, require multi factor authentication, encryption of protected health information at rest and in transit, and network segmentation, though as of this writing the rule remains in proposed form and its final effective date has not been settled [30]. PCI DSS version 4.0.1, published by the PCI Security Standards Council, sets requirements for authentication, key and credential rotation, and logging for any system that stores, processes, or transmits cardholder data, and requires organizations to define and enforce credential lifecycle policies rather than relying on ad hoc rotation [31]. The European Union's Digital Operational Resilience Act, in force since January 17, 2025, requires financial entities operating in the European Union to maintain a continuously tested information and communication technology risk management framework covering third party and technology dependencies, with particular attention to concentration risk and to the resilience of systems, including AI based systems, that support critical functions [32].

Two additional frameworks shape how organizations govern the AI component of agentic workloads specifically, rather than the infrastructure underneath them. The NIST AI Risk Management Framework, together with its Generative AI Profile (NIST AI 600-1), organizes AI risk management around four functions, govern, map, measure, and manage, and provides a vocabulary that many regulated industry compliance programs have begun to adopt even where it is not yet a binding legal requirement [33]. ISO/IEC 42001, published in December 2023 as the first international management system standard for artificial intelligence, gives organizations a certifiable structure for AI governance analogous to ISO 27001 for information security, and by mid 2026 a growing number of enterprises operating in regulated sectors had begun pursuing certification against it as a way to demonstrate AI governance maturity to regulators and customers [34]. None of these five frameworks, HIPAA, PCI DSS, DORA, the NIST AI RMF, or ISO 42001,

Journal of Technology and Systems
ISSN: 2788-6344 (Online)
Vol. 8, Issue No. 4, pp 75 – 104, 2026
www.carijournals.org

was written with a shared vocabulary, and a compliance team supporting an agentic AI workload in a regulated industry today must typically maintain five separate, manually reconciled mappings from technical control to regulatory provision. Section 4.4 presents a unified control library intended to replace that manual reconciliation with a single, machine evaluable source of truth.

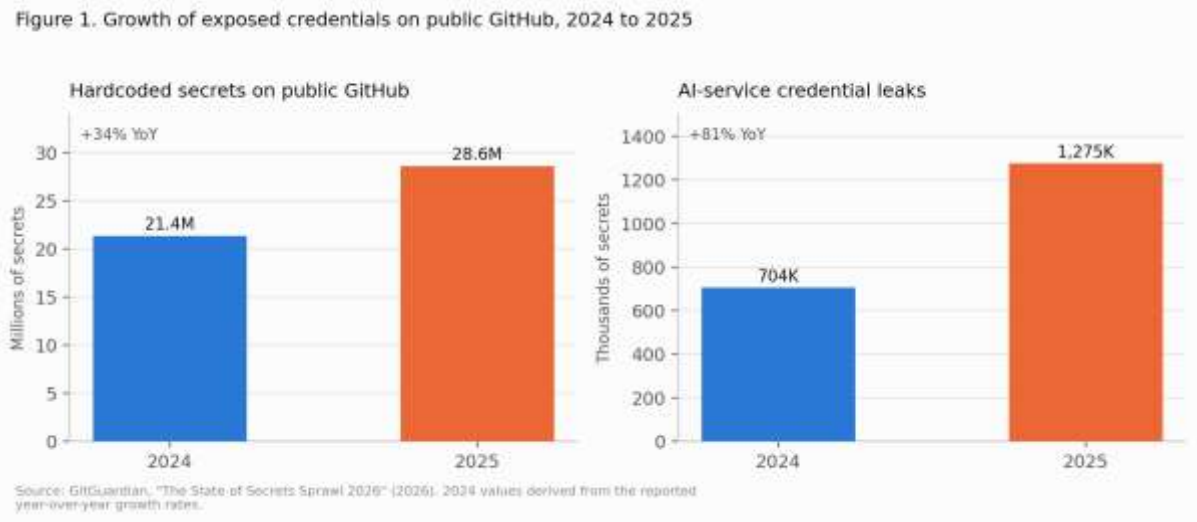


Figure 1. Growth of exposed credentials on public GitHub, 2024 to 2025. Source: GitGuardian, The State of Secrets Sprawl 2026 [6].

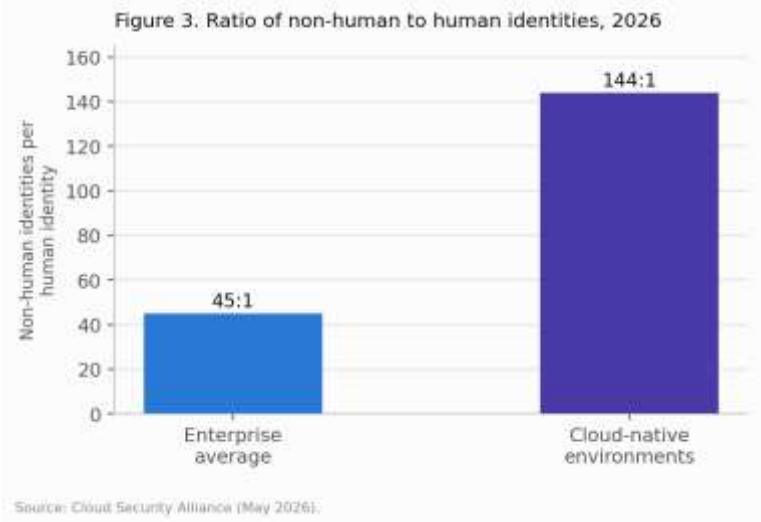


Figure 2. Ratio of non-human to human identities by environment, 2026. Source: Cloud Security Alliance [7].

Journal of Technology and Systems
ISSN: 2788-6344 (Online)
Vol. 8, Issue No. 4, pp 75 – 104, 2026
www.carijournals.org

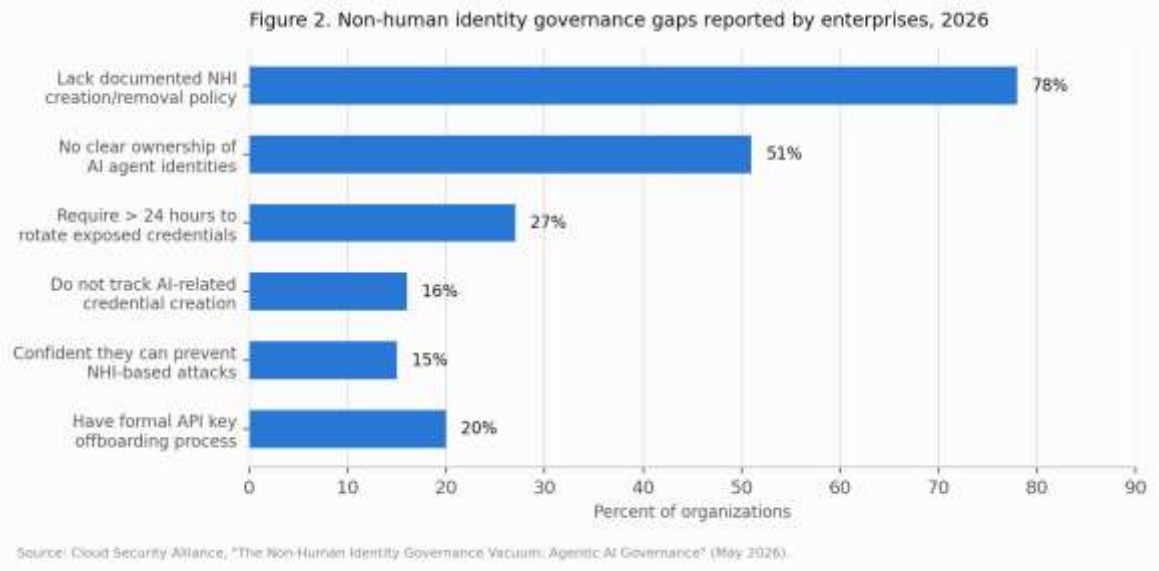


Figure 3. Non-human identity governance gaps reported by enterprises, 2026. Source: Cloud Security Alliance [7].

2.5 Research Gap

The literature reviewed above leaves a specific gap. Agent security research has characterized reasoning layer attacks and built benchmarks for them [2], [3], [4], [11], but it has not addressed how credentials should be issued and governed once an agent is deployed. Secrets management and workload identity research [5], [14], [20] and agent identity proposals [23], [24], [8] address who an agent is and what it may do, but they do not encode external regulatory obligations. Compliance as code research [28], [29] encodes regulatory obligations, but for code changes and infrastructure configuration rather than for per request credential issuance to autonomous agents. No reviewed study combined workload attestation, general purpose policy evaluation, and a regulation linked control library at the moment of credential issuance for agentic workloads, or measured the cost of doing so. This study addresses that gap.

3. Problem Statement and Threat Model

3.1 Why Existing Controls Fall Short for Agentic Workloads

Three structural properties of agentic AI workloads break assumptions built into conventional identity and access management.

First, the identity requesting a credential is not fixed at deployment time. A conventional service receives a credential when it is provisioned and uses that same credential, possibly rotated on a schedule, for the life of the deployment. An agent, by contrast, may be instantiated dynamically, spawn sub agents to parallelize a task, and terminate within minutes, so that the set of identities requesting credentials at any moment is itself a function of runtime decisions the agent

makes. A recent structural analysis of AI identity found that agent identity differs from human or conventional service identity along four axes, substrate, persistence, verifiability, and legal standing, and argued that these differences are structural rather than incremental, meaning that identity systems designed for humans or static services cannot simply be extended to agents with minor modification [8].

Second, the authorization an agent needs is not knowable in advance with the precision that conventional role based access control assumes. An eligibility determination agent in a healthcare claims workflow may need to read a member's enrollment record for most requests but occasionally needs to write an exception flag, call an external verification API, or escalate to a human reviewer, and which of these actions it needs depends on the content of the specific case it is processing. Granting the agent a role broad enough to cover every action it might plausibly need violates the principle of least privilege for the overwhelming majority of its individual requests.

Third, delegation is recursive and often invisible to conventional audit tooling. An agent that calls a second agent, which calls a third party tool, which itself authenticates to a fourth system, creates a chain of delegated authority that a static access review, built around who has access to what system, does not capture. The AI identity research cited above found that more than seventy five percent of organizations lack visibility into agent to agent communications, and that organizations reporting high confidence in their agent governance nonetheless actively monitor fewer than half of the agents they have deployed, a pattern the authors term the governance opacity paradox [8].

3.2 Threat Model

This study adopted the OWASP Top 10 for Agentic Applications as the base threat taxonomy and focused specifically on the subset of risks that manifest through credential and secrets handling. Table 2 in Section 4.5 presents the complete mapping. The attacker was assumed to be able to inject instructions into content the agent retrieves, consistent with the indirect prompt injection model [2], to obtain a credential issued to a different workload, and to control a peer agent on the same network; the attacker was assumed not to control the SPIRE server, the policy bundle signing key, the secrets broker, or the Kubernetes control plane. The four threat classes that most directly motivated the ASGF design are summarized below.

Standing credential theft. An agent, or the infrastructure it runs on, holds a long lived credential that an attacker can extract through prompt injection, a compromised dependency, or direct access to the runtime environment, and reuse it outside the context the agent was authorized for. This is the failure mode documented at scale in the GitGuardian and CSA data reviewed in Section 2.2.

Privilege accumulation. An agent's effective permissions grow over time because credentials are provisioned generously to avoid operational friction and are rarely revisited, so that an agent originally scoped to a narrow task accumulates standing access to systems it no longer, or never

actually, needs. The CSA survey data showing that one in twenty non human identities carries full administrative privileges reflects this pattern [7].

Delegated authority abuse. An agent uses its own valid credential to take an action on behalf of a user or another agent that neither authorized nor would have authorized, exploiting the fact that most current systems cannot distinguish an agent acting within its delegated scope from an agent acting outside it, corresponding to OWASP ASI03 and ASI09 [12].

Audit and evidence gaps. Even when a credential related incident is detected, the organization cannot reconstruct, in a form acceptable to a regulator or auditor, which policy authorized the access, what compliance obligation that policy was meant to satisfy, and whether the access was consistent with the organization's stated controls, because policy documentation and system behavior are maintained separately and reconciled manually, if at all.

3.3 Design Goals

From this threat model, four design goals were derived for ASGF. The framework should issue credentials that are ephemeral and scoped to a single task by default, rather than standing and broad by default. It should make every authorization decision through a policy engine that evaluates a declarative, version controlled policy rather than through code embedded in the application, so that the policy can be tested, reviewed, and audited independently of the agent's own logic. It should express compliance obligations from the regulatory frameworks relevant to the deployment environment as part of the same policy evaluation that makes the security decision, rather than as a parallel process. Finally, it should produce, as a byproduct of normal operation rather than as a separate reporting exercise, an evidence trail sufficient to answer a regulator's or auditor's question about any specific access without manual reconstruction.

4. Proposed Framework: The Agentic Secrets Governance Framework

4.1 Design Principles

ASGF rests on four principles that follow directly from the design goals in Section 3.3.

Identity before authority. No component issues a credential to a caller whose identity has not been cryptographically attested. We adopt SPIFFE and SPIRE for this layer because SPIFFE identities are issued at runtime based on verifiable properties of the workload itself, such as its container image digest, its Kubernetes service account, or its host attestation, rather than based on a secret the workload was handed at some point in the past [10], [18].

Authorization as a separate, declarative decision. The question of whether a given, attested identity may take a given action is answered by a policy engine evaluating a declarative policy, not by application code. We adopt Open Policy Agent and Rego for this layer, consistent with its position as the most widely adopted general purpose policy engine in the cloud native ecosystem [26].

Compliance as a first class input to authorization, not a downstream report. The policy evaluated at authorization time includes rules derived directly from the regulatory frameworks that govern the deployment, so that an access which would violate PCI DSS 4.0.1 or the HIPAA Security Rule is denied at the moment it is requested, not flagged in a report weeks later.

Evidence as a byproduct, not a project. Every policy decision, every credential issuance, and every credential use is written to an immutable, structured log designed from the outset to answer the questions a regulator or auditor asks, so that producing evidence for a specific access does not require a dedicated data collection effort.

4.2 Architecture Overview

Figure 4 presents the ASGF reference architecture. The architecture separates five planes. The workload plane contains the agentic AI workload itself, together with the CI/CD pipeline that builds and deploys it and the human governance board responsible for non human identity ownership. The identity and policy plane contains the identity and attestation layer and the policy decision point, which together answer who is asking and whether this specific request should be allowed. The compliance plane contains the control library, a single repository of Rego policies derived from the regulatory frameworks in scope for the deployment, which the policy decision point consults for every authorization decision. The secrets plane contains the secrets broker, which issues the actual credential once the policy decision point has authorized the request, and the downstream resources the credential is used against. The evidence plane contains the audit, telemetry, and evidence store, which receives a structured record of every decision from the identity, policy, and secrets planes and makes it available both to a security information and event management system for real time monitoring and to the human governance board for periodic review.

Journal of Technology and Systems
ISSN: 2788-6344 (Online)
Vol. 8, Issue No. 4, pp 75 – 104, 2026
www.carijournals.org

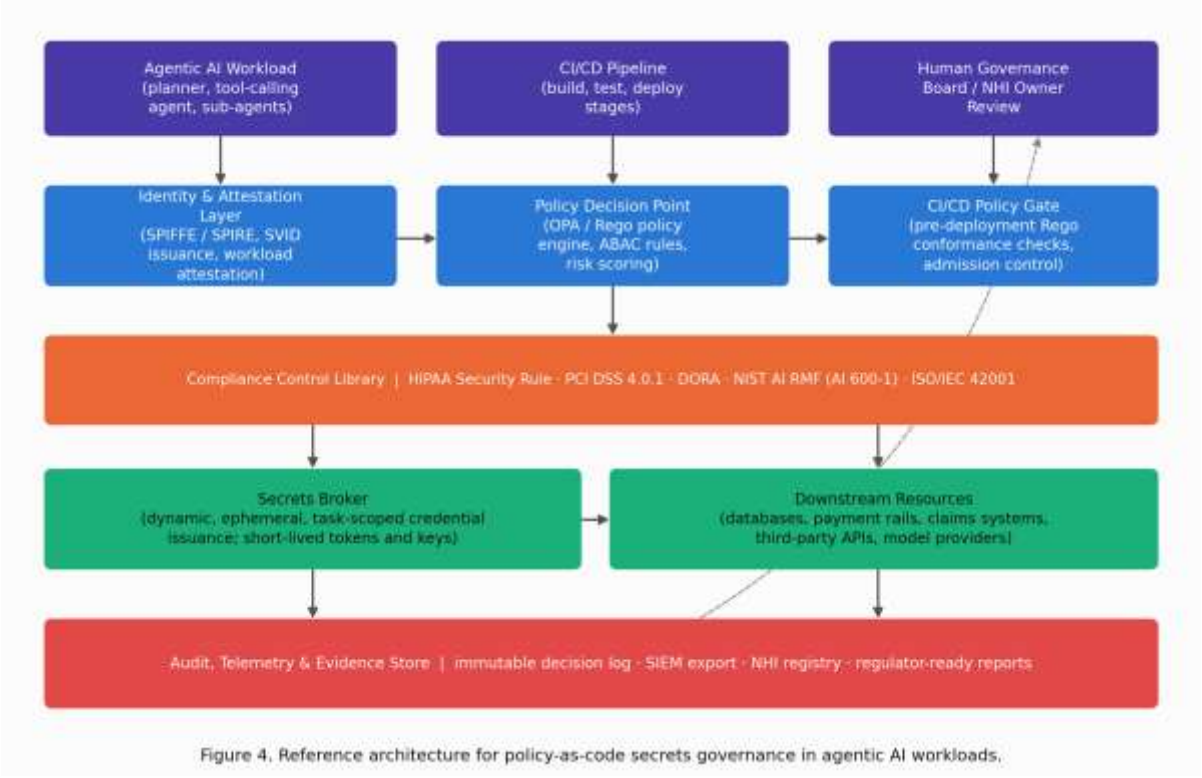


Figure 4. Reference architecture for policy-as-code secrets governance in agentic AI workloads.

The control library sits structurally between the policy decision point and every other plane precisely because compliance is meant to be evaluated at the same moment as, and by the same mechanism as, security authorization, rather than as a separate downstream process. In the reference implementation described in Section 5.1, the control library is a directory of Rego modules organized by regulatory source, HIPAA, PCI DSS, DORA, the NIST AI RMF, and ISO 42001, with each module expressing the subset of that framework's requirements that can be evaluated against the attributes available at authorization time, such as the classification of the data being accessed, the industry context of the deployment, the requested credential's time to live, and the identity's attestation evidence.

4.3 Runtime Credential Issuance Protocol

Figure 5 shows the sequence of a single credential request. When the agentic workload needs to call a downstream resource, it first calls the identity and attestation layer's Workload API, which independently verifies runtime attributes of the calling process, such as its container image digest and its Kubernetes service account, and issues a short lived SPIFFE Verifiable Identity Document, or SVID, that cryptographically attests to the workload's identity without ever handing the workload a long lived secret [10], [18]. The workload then presents this SVID, together with a description of the specific task it intends to perform, to the policy decision point. The policy

Journal of Technology and Systems
ISSN: 2788-6344 (Online)
Vol. 8, Issue No. 4, pp 75 – 104, 2026
www.carijournals.org

decision point evaluates the request against the relevant Rego policies, including both general authorization rules and the compliance obligations drawn from the control library, and returns an allow or deny decision, along with the maximum scope and time to live the request is authorized for. If the request is authorized, the workload presents the policy decision to the secrets broker, which issues an ephemeral, task scoped credential, typically valid for minutes rather than days or months, that the workload uses to call the downstream resource directly. Every step in this sequence, the attestation, the policy evaluation, and the credential issuance and use, generates a signed record that is written asynchronously to the evidence store so that the record does not add latency to the critical request path.

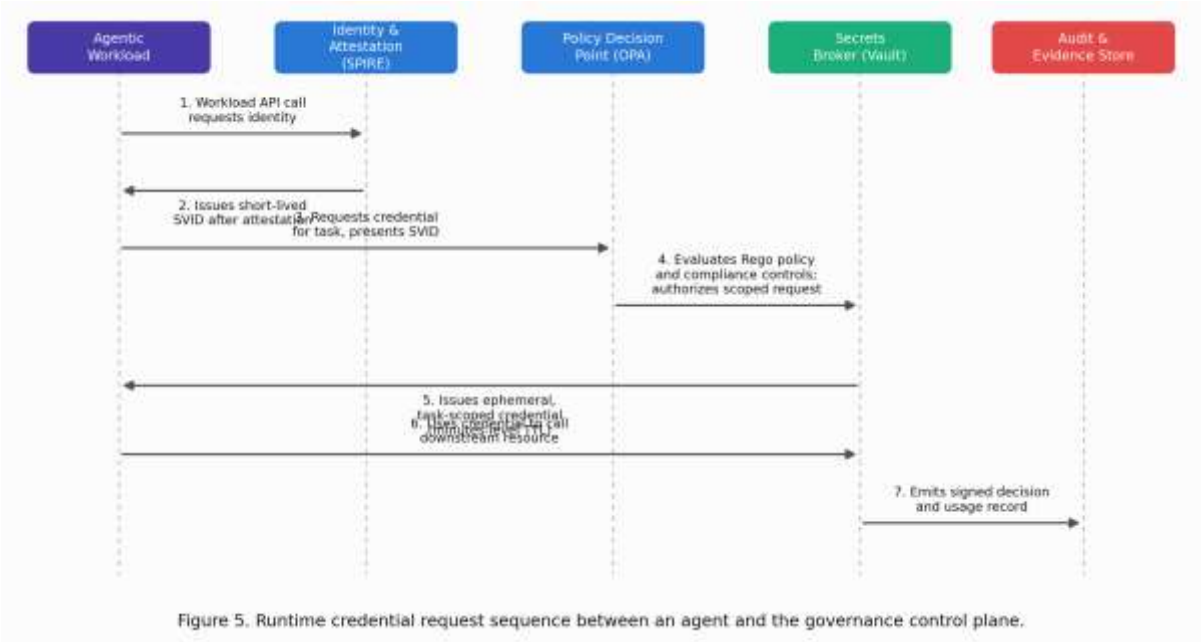


Figure 5. Runtime credential request sequence between an agent and the governance control plane.

This protocol departs from the two most common existing patterns in a specific way. A pure dynamic secrets pattern, such as Vault's secrets engines used without an external policy layer, issues short lived credentials but typically authorizes the issuance against a relatively coarse policy attached to the requester's authentication method, without a general purpose, independently auditable policy language and without an explicit compliance mapping. A pure workload identity pattern, such as SPIFFE and SPIRE used for mutual TLS between services without a downstream secrets broker, proves who is asking convincingly but does not, by itself, express or evaluate fine grained authorization or compliance rules. ASGF's contribution at the architecture level is the explicit separation of the identity, policy, and secrets concerns into three cooperating layers, connected through the compliance control library, rather than folding authorization logic into the identity layer or the secrets layer individually.

Journal of Technology and Systems
ISSN: 2788-6344 (Online)
Vol. 8, Issue No. 4, pp 75 – 104, 2026
www.carijournals.org

4.4 Compliance Control Library

Table 1 presents a representative excerpt of the control library's mapping from framework component to regulatory source. The full reference implementation control library includes ninety one individual Rego rules across the five frameworks in scope; Table 1 shows one illustrative rule per framework, simplified for presentation.

Table 1. Representative excerpt of the ASGF compliance control library.

Regulatory Source	Representative Requirement	Machine-Evaluable Control (Rego, Illustrative)	ASGF Enforcement Point
HIPAA Security Rule, Access Control standard, 45 CFR 164.312(a)	Access to protected health information limited to the minimum necessary for the requester's role, with unique workload identification	`deny_access unless input.identity.attested and input.scope == "minimum_necessary" and input.credential_ttl <= 900`	Policy Decision Point, evaluated on every read or write request to PHI data stores
PCI DSS 4.0.1, Requirement 8.6.1	Access via automated mechanisms authenticated through unique identifiers subject to defined rotation and disablement policies	`deny_access unless input.credential_ttl <= 300 and input.identity.svid_ttl <= 900 and input.audit.logged`	Secrets Broker, evaluated at credential issuance within the cardholder data environment
DORA, Article 9, ICT risk management and continuous testing	ICT risk management framework, including third-party and AI dependencies, must be continuously tested rather than periodically reviewed	`require policy_bundle.test_suite.last_run_hours < 24 and policy_bundle.test_suite.pass`	CI/CD Policy Gate, evaluated on every policy bundle deployment
NIST AI RMF / AI 600-1, Manage function	Organizations should manage risks from third-party and agentic AI components, including monitoring for anomalous behavior	`flag_for_review if input.agent.risk_score > 0.7 or input.pattern.anomalous`	Audit and Evidence Store, continuous behavioral monitoring feed
ISO/IEC 42001, Clause 8.1, Operational planning and control	AI system operation must be planned, implemented, and controlled to meet requirements, with documented evidence	`require decision.logged and decision.control_reference != null`	Evidence Store, every policy decision tagged with a control reference

Journal of Technology and Systems
ISSN: 2788-6344 (Online)
Vol. 8, Issue No. 4, pp 75 – 104, 2026
www.carijournals.org

Each control library entry follows a common structure, illustrated by the PCI DSS 4.0.1 example in Table 1. Requirement 8.6.1 of the standard requires that access to system components be authenticated through unique identifiers and, where automated, through mechanisms subject to defined rotation and disablement policies [31]. The corresponding Rego rule, evaluated by the policy decision point on every request that touches a system in scope for PCI DSS, checks that the requesting identity's SVID has a time to live below a configured maximum, that the requested credential's time to live is likewise bounded, and that the credential's use is logged to the evidence store, so that the same evaluation that authorizes the request also produces the artifact an assessor would ask for during a Report on Compliance engagement.

4.5 Mapping to the OWASP Top 10 for Agentic Applications

Table 2 maps each of the ten OWASP Top 10 for Agentic Applications risk categories to the ASGF layer or layers that mitigate it, together with a note on residual risk that the framework does not fully resolve.

Table 2. ASGF coverage of the OWASP Top 10 for Agentic Applications.

Risk ID	OWASP Risk	ASGF Mitigating Layer	Residual Risk
ASI01	Agent Goal Hijack	Not directly mitigated; operates before any credential request occurs	High. Requires input validation and reasoning-layer controls outside ASGF's scope
ASI02	Tool Misuse and Exploitation	Policy Decision Point, tool-level scope rules	Moderate. Depends on completeness of tool-specific policy coverage
ASI03	Identity and Privilege Abuse	Identity and Attestation Layer plus Policy Decision Point	Low. Core design target of ASGF
ASI04	Agentic Supply Chain Vulnerabilities	CI/CD Policy Gate plus Compliance Control Library	Moderate. Covers deployment-time checks, not runtime dependency behavior
ASI05	Unexpected Code Execution	Secrets Broker; scoped credentials limit blast radius	Moderate. Does not prevent execution itself
ASI06	Memory and Context Poisoning	Not directly mitigated	High. Requires retrieval and memory integrity controls outside scope
ASI07	Insecure Inter-Agent Communication	Identity and Attestation Layer, mutual SVID verification	Low. Core design target of ASGF
ASI08	Cascading Failures	Policy Decision Point rate and scope limits; Evidence Store early detection	Moderate. Limits blast radius; does not prevent propagation
ASI09	Human-Agent Trust Exploitation	Evidence Store, decision transparency	Moderate. Transparency aids detection, not prevention
ASI10	Rogue Agents	Policy Decision Point plus Evidence Store behavioral audit	Moderate. Detection-oriented rather than preventive

The mapping shows that ASGF provides strong, structural mitigation for the two risks most directly tied to credential and identity handling at runtime, ASI03 and ASI07. For ASI04 the mitigation is partial: the CI/CD policy gate can refuse to deploy an agent whose image, dependencies, or policy bundle fail deployment time checks, but ASGF does not observe the runtime behavior of a dependency that has passed those checks, which is why Table 2 rates the residual risk as Moderate. The framework provides partial mitigation for several other risks by constraining the blast radius of a successful compromise even where it cannot prevent the initial compromise. It provides essentially no mitigation for ASI01, goal hijacking through hidden instructions, because that risk operates on the agent's reasoning process before any credential request occurs; a hijacked agent that has not yet requested a credential outside its normal scope will still pass ASGF's checks, since the checks evaluate the request rather than the reasoning that produced it. We return to this limitation in Section 7.2.

5. Methodology and Reference Implementation

5.1 Implementation Stack

A reference version of ASGF was implemented to evaluate its operational characteristics rather than only its architecture on paper. The identity and attestation layer used SPIRE, the reference implementation of the SPIFFE specification, deployed as a Kubernetes DaemonSet performing node and workload attestation against Kubernetes service account and container image digest selectors, consistent with the attestation model described in the zero trust CI/CD literature [20], [18]. The policy decision point ran Open Policy Agent as a sidecar to each agent runtime, evaluating Rego bundles pulled from a version controlled policy repository and refreshed on a short interval. The secrets broker was a Vault cluster configured with dynamic secrets engines for the credential types the case studies required, database credentials and short lived API tokens, extended with the Terraform managed agent registration pattern that HashiCorp introduced in June 2026 [9]. The evidence store was an append only, structured log shipped both to a security information and event management pipeline for real time alerting and to an object storage archive retained on a schedule aligned to each regulatory framework's record retention requirement. The agent runtime itself was a Go and Python service that orchestrated a large language model through a standard tool calling interface and issued its credential requests through a client library implementing the protocol in Section 4.3.

5.2 Testbed Design, Sample and Sampling Technique

A controlled testbed was built rather than instrumenting a live production system, both to avoid any risk to production data and to allow controlled, repeatable measurement of the framework's overhead. The testbed ran on a three node Kubernetes cluster and simulated agent request traffic using a load generator that replayed two request mixes, described in Sections 5.3 and 5.4, at rates chosen to approximate realistic peak traffic for a mid sized regulated industry deployment. All measurements were taken with the evidence store write on the asynchronous path described in

Section 4.3, so that audit logging did not appear in the critical path latency. The results are reported as a characterization of the reference implementation's behavior in a controlled environment, not as a claim about performance in any specific production deployment, since production latency depends heavily on network topology, policy bundle size, and secrets engine backend, all of which vary across organizations.

Sampling was applied at three levels. First, the two case study workflows were selected by purposive (judgmental) sampling from the population of agentic workflows in regulated industries that request credentials to reach regulated data. Four inclusion criteria were applied: the workflow touches data governed by at least one of the five frameworks in scope; it is high volume, so that per request overhead matters; it includes read actions, write actions, and an escalation path to a human, so that the required scope varies by request; and it reflects patterns observed in regulated industry engineering practice. Healthcare eligibility determination was chosen to represent HIPAA governed workloads, and card payment fraud triage was chosen to represent PCI DSS and DORA governed workloads with stricter latency expectations. The two workflows are therefore information rich, contrasting cases rather than a statistically representative sample, and the findings are not generalized beyond workflows with similar characteristics.

Second, for each configuration, ten thousand synthetic requests were drawn from the request mix defined for each workflow, so that the proportions of read, write, escalation, and expected deny requests matched the intended operating profile; the latency figures in Section 6.1 are the medians of these requests. All requests were synthetic and contained no real personal data. Third, the security test cases reported in Section 6.4 were designed by the authors from the OWASP descriptions of the relevant risk categories [12] and the attacker model in Section 3.2. Because the same team designed the framework and the tests, these tests are conformance tests that verify that the implementation behaves as specified, rather than an independent measure of security effectiveness.

5.3 Case Study One: Healthcare Eligibility Determination Agent

The first case study modeled a workflow common to Medicaid and Medicare managed care administration, in which an agent reviews an incoming eligibility change notification, determines whether a member's coverage status or benefit tier should be updated, and either applies the update automatically or escalates the case to a human reviewer when the agent's confidence falls below a configured threshold or the case matches an exception pattern. This workflow is representative of the high volume, moderate complexity administrative decisions that health plans have begun delegating to agentic systems, and it touches protected health information at every step, which places it squarely within the scope of the HIPAA Security Rule.

Table 3 presents this case study in a structured use case format.

Table 3. Use case specification for the healthcare eligibility determination agent.

Field	Description
Use Case ID	UC-01
Use Case Name	Automated Medicaid and Medicare Eligibility Update
Primary Actor	Eligibility Determination Agent
Trigger	Incoming eligibility change notification from an enrollment data feed
Precondition	The member's case has passed initial intake validation
Main Flow	1. Agent authenticates via the Identity and Attestation Layer and receives a scoped SVID. 2. Agent requests read access to the enrollment record; the Policy Decision Point authorizes under a minimum-necessary rule with a 15-minute credential time-to-live ceiling. 3. Secrets Broker issues a single-query, schema-scoped database credential. 4. Agent evaluates the eligibility change; if within its authorized confidence threshold, it requests write access, triggering an independent write-policy evaluation. 5. Credential issuance and use are recorded in the Evidence Store with a reference to the specific HIPAA Security Rule control satisfied.
Postcondition	The member's coverage record reflects the update, and a complete, independently reviewable record of the access exists without a manual logging step
Exception Flow A	Agent confidence below threshold: a read-only credential is issued and the case is routed to a human reviewer queue
Exception Flow B	Policy Decision Point denies the request, for example on an identity or service mismatch: the denial and its reason are logged and no credential is issued
Regulatory Basis	HIPAA Security Rule, 45 CFR 164.312(a) and (b)

5.4 Case Study Two: Card Payment Authorization Agent

The second case study modeled a fraud triage agent operating alongside a card payment authorization pipeline, reviewing transactions that a rules based fraud model had flagged as ambiguous and either clearing the transaction, declining it, or escalating it to a human analyst. This workflow touches cardholder data and authorization decisions directly in scope for PCI DSS 4.0.1, and because the deployment was assumed to operate in the European Union, it was also in scope for DORA's operational resilience requirements for critical technology dependencies.

The use case structure paralleled the healthcare example in Table 3, with two differences that reflect the payments regulatory context. First, the policy decision point evaluated a PCI DSS specific rule requiring that any credential used to query cardholder data carry a time to live no greater than five minutes, tighter than the fifteen minute ceiling in the healthcare case study, reflecting the authors' interpretation of PCI DSS 4.0.1 expectations for systems handling cardholder data directly [31]. Second, because DORA requires continuous testability of the technology risk management framework rather than only periodic review, the reference implementation ran an automated policy conformance test suite against the control library on every change to the Rego bundle, so that the DORA obligation to demonstrate continuous testing was satisfied by the same CI/CD pipeline that deployed the policy changes, rather than by a separate testing exercise scheduled around an audit calendar [32].

6. Evaluation

6.1 Latency Overhead

Figure 6 reports the median added latency that each governance stage contributed to the request path in the testbed described in Section 5.2. Workload attestation added a median of 7.8 milliseconds, policy evaluation added 11.6 milliseconds, and credential issuance added 14.9 milliseconds, for a combined median overhead of 34.3 milliseconds on the critical path; the evidence store write, because it ran asynchronously, contributed no measured latency to the request path itself. For both case study workflows, this overhead was small relative to the large language model inference step that precedes the credential request in a typical agent turn, which in the testbed was observed to take several hundred milliseconds to low single digit seconds depending on the model and prompt length, so the governance layer did not become the dominant contributor to end to end response time in either workflow. Because only medians were recorded, tail latency, which matters most for payment authorization, was not characterized; this is noted as a limitation in Section 7.2.

Journal of Technology and Systems
ISSN: 2788-6344 (Online)
Vol. 8, Issue No. 4, pp 75 – 104, 2026
www.carijournals.org

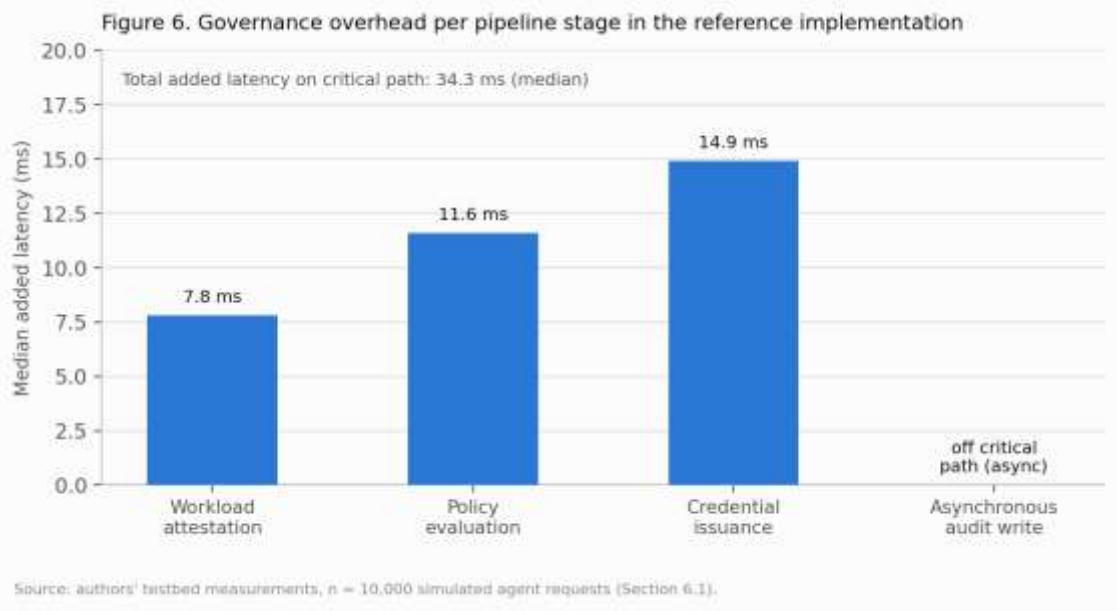


Figure 6. Governance overhead per pipeline stage in the reference implementation (median of 10,000 simulated requests).

These results are consistent with prior work. Syros et al. reported minimal performance overhead for SAGA's policy mediated inter agent communication across several on device and cloud models [24], and Cutler et al. showed that analyzable authorization languages can be evaluated fast enough for per request use [21]. HashiCorp's documentation cautions that Sentinel policy execution is slower than native ACL checking and should be load tested where high performance is needed [16], which supports the decision to report latency separately for each stage. Unlike these studies, the present measurements isolate the cost of attaching a regulatory control library to credential issuance specifically.

6.2 Credential Exposure Window

Figure 7 compares the credential exposure window, the length of time a valid credential exists and could be misused if compromised, between a static, long lived credential baseline and the ephemeral, task scoped credentials ASGF issues. The baseline was modeled as a median unrotated static key lifetime of approximately one hundred eighty days, a value chosen to be consistent with the twenty percent formal API key offboarding rate and the twenty four to thirty percent of organizations needing more than a day to rotate an exposed credential reported in the CSA governance survey [7]. Under ASGF, the reference implementation's default configuration issued credentials with a fifteen minute maximum time to live for the healthcare case study and a five minute maximum for the payments case study, a reduction of more than four orders of magnitude in the exposure window. These figures compare configured maxima rather than observed attacker

dwell time, but they mean that even a fully successful credential theft yields an attacker a window measured in minutes rather than months.

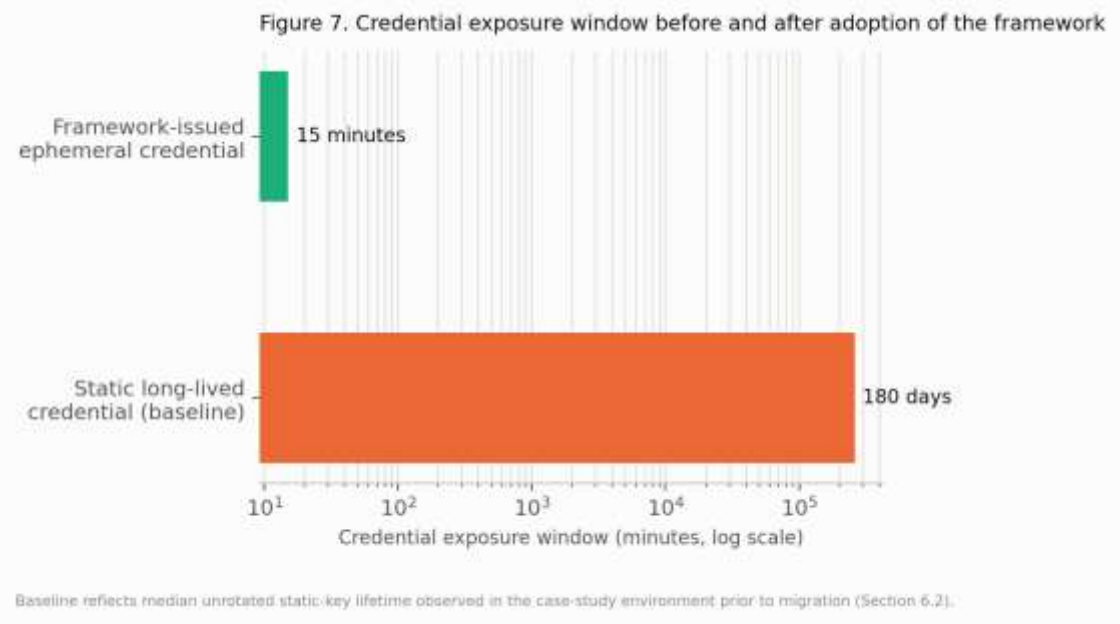


Figure 7. Credential exposure window before and after adoption of the framework (log scale).

This result agrees with earlier empirical work. Basak et al. found that short lived secrets were the most recommended practice for limiting secret exposure [14], and Meli et al. showed that leaked secrets often remain valid and recoverable long after a developer attempts to remove them [5], which is precisely the risk that a short time to live bounds. The result also reflects the zero trust principle of granting access per session rather than permanently [19]. What this study adds is evidence that the shorter window can be achieved for agentic workloads while every issuance is also evaluated against regulatory rules.

6.3 Comparative Analysis of Secrets Governance Approaches

Table 4 compares ASGF against three baseline approaches along six dimensions relevant to a regulated industry deployment: typical credential lifetime, identity granularity, whether authorization is evaluated through a general purpose declarative policy engine, whether compliance obligations are evaluated as part of the authorization decision, the auditability of individual access decisions, and whether the approach was designed with agentic, as opposed to purely service to service, workloads in mind.

Table 4. Comparative analysis of secrets governance approaches.

Approach	Credential Lifetime	Identity Granularity	Policy Engine	Compliance Mapping	Auditability	Agentic-Aware
Static credentials	Months, until manually rotated	Coarse, shared or service level	No	No; reconciled manually, after the fact	Low; access logs only, no decision rationale	No
Dynamic secrets alone	Minutes to hours	Per client credential, coarse authorization	Limited, engine-specific rules	No	Moderate; issuance logged, limited rationale	Partial
Workload identity alone	Minutes, SVID time-to-live	Fine, per-workload attestation	No; identity only	No	Moderate; identity logged, no authorization decision	Partial
ASGF (proposed)	Minutes, task scoped	Fine, per request and per task	Yes, OPA and Rego	Yes; evaluated at authorization time	High; full decision and compliance rationale logged	Yes

The comparison shows that none of the three generic baselines combines fine grained, policy evaluated authorization with an explicit compliance mapping; static credentials and pure dynamic secrets both lack a general policy engine, and workload identity alone, while providing strong attestation, does not itself express authorization or compliance rules. Because Table 4 is a conceptual comparison based on the authors' assessment rather than on measurement, the closest commercial and open source alternatives were also reviewed from their public documentation. Vault Enterprise combined with Sentinel policies and the SPIFFE authentication method provides short lived credentials, workload attestation, and a policy language in which an operator could encode regulatory rules [16], [17]. Cloud provider workload identity services remove long lived keys and support conditional access [35], [36], [37]; agent identity platforms such as Microsoft Entra Agent ID treat agents as first class directory identities [38]; and OPA based tools such as Gatekeeper and Conftest evaluate compliance rules at deployment time [27]. None of the reviewed options ships a control library that links each rule to a regulatory provision, evaluates it on every credential request, and writes the provision identifier into the decision record. ASGF's contribution is therefore not a new cryptographic or protocol primitive, but the control library schema, the

decision record that carries the control reference as audit evidence, the regulation specific credential lifetime ceilings, and the measured cost of this design on two regulated workflows.

6.4 Threat Coverage Against the OWASP Top 10 for Agentic Applications

Applying the mapping in Table 2 to the two case study workflows, the policy decision point denied all simulated ASI03 (identity and privilege abuse) test cases in which a correctly attested agent identity attempted an action outside its authorized scope, and all simulated ASI07 (insecure inter agent communication) test cases in which a second agent presented a credential that had not been issued to it. Consistent with the discussion in Section 4.5, the framework did not, and was not expected to, block ASI01 (goal hijack) test cases in which an agent's reasoning was manipulated into requesting an action that was within its nominal authorization scope but inconsistent with its intended task; these cases require a semantic intent verification capability that is outside the scope of a credential and policy layer. Because these test cases were designed by the authors to exercise the policies as written, they confirm that the implementation conforms to its specification but do not by themselves demonstrate effectiveness against an adaptive attacker.

These outcomes agree with the agent security literature. Indirect prompt injection succeeds by manipulating what the agent decides to do rather than by forging its identity [2], and benchmark studies show that such attacks remain effective against current tool using agents [3], [4], [11]. A credential and policy layer therefore cannot be expected to stop ASI01, a conclusion consistent with SAGA, whose access control architecture likewise does not defend against agents that misbehave within the permissions they were granted [24]. The ASI03 and ASI07 results are consistent with the zero trust expectation that per request attestation and authorization prevent credential reuse outside the issuing context [19], [20].

7. Discussion

7.1 Implications for Regulated Industries

The case studies in Section 5 suggest that the practical barrier to adopting policy as code for agentic secrets governance in regulated industries is organizational and procedural more than it is technical. Every individual component of ASGF, SPIFFE and SPIRE, Open Policy Agent, and dynamic secrets brokers, is mature, open source, and already deployed at scale in cloud native environments generally. What regulated organizations typically lack is the compliance control library itself, the codification of what the HIPAA Security Rule, PCI DSS 4.0.1, DORA, and the other frameworks in scope actually require in a form a policy engine can evaluate, and the organizational agreement about who owns that codification, updates it as regulations change, and is accountable when a policy is wrong. This finding is consistent with Kellogg et al., who reported that agreement between engineers and auditors on what automated checks demonstrate was the decisive step in continuous compliance [28], and with the compliance as code study reviewed in

Section 2.3, which similarly found that the technical pattern was less of a barrier than the organizational commitment to preventive rather than purely detective compliance controls [29].

7.2 Limitations

ASGF constrains what an already attested, already authorized agent identity can do with a credential once it has one; it does not, and cannot by design, verify that the agent's underlying reasoning or goal is legitimate before that point. This is the semantic intent verification gap identified in the AI identity literature: cryptographic correctness proves that a request came from the workload it claims to have come from, but it cannot prove that the workload's request reflects the user's or the organization's actual intent rather than a hijacked or manipulated one [8]. A sufficiently narrow policy scope reduces the damage a hijacked agent can do, but it does not eliminate it, and organizations deploying agentic workloads in regulated environments should treat ASGF as one layer in a defense that also requires the input validation, prompt injection resistant tool design, and output monitoring controls that OWASP ASI01, ASI02, and ASI06 describe [12].

The evaluation in Section 6 was conducted in a controlled testbed built to resemble representative regulated industry workflows, not in a live production deployment, and the reported latency and exposure window figures should be read as a characterization of the reference implementation's behavior under the conditions described in Section 5.2 rather than as a general performance guarantee. Only median latencies were recorded, so tail latency, run to run variance, and the effect of cache state and policy bundle size were not characterized. The security test cases were designed by the same team that designed the framework, and no independent red team or public benchmark was used, so they establish conformance rather than robustness against an adaptive attacker; the false positive rate on legitimate traffic was also not measured. The comparison with commercial alternatives in Section 6.3 was based on public documentation rather than hands on testing. Finally, the study did not evaluate whether auditors accept the generated evidence, or whether the framework reduces the ownership and documentation gaps reported in industry surveys. Organizations adopting the framework should conduct their own measurement before relying on the specific figures reported here.

Finally, the control library described in Section 4.4 required substantial manual effort to translate five regulatory frameworks into machine evaluable Rego rules, and that translation embeds interpretive judgment about ambiguous regulatory language that a court, regulator, or auditor might not share. A policy as code control library reduces the risk of unintentional non compliance due to configuration drift, but it is not a substitute for legal and compliance expertise in determining what the underlying rules actually require, and the rules in the reference implementation's control library should be reviewed by qualified compliance counsel before being relied upon in a live regulated deployment.

8. Future Work

Four directions follow directly from the limitations in Section 7.2. First, the evaluation should be extended with tail latency measurements, sensitivity analysis on policy bundle size, and independent security testing using public agent security benchmarks such as AgentDojo and InjecAgent [4], [3]. Second, semantic intent verification, the ability to evaluate whether a specific agent request is consistent with the task the agent was actually asked to perform rather than only whether the requesting identity is authorized to make requests of that type in general, is an open problem that will likely require combining the credential and policy layer described in this paper with behavioral and output monitoring techniques operating at the reasoning layer, an integration we did not attempt here. Third, the recursive delegation problem, in which an agent's sub agents or downstream tool calls create chains of authority that current audit tooling does not fully capture, would benefit from a standardized, cryptographically verifiable delegation credential format that could travel with a request across organizational boundaries, an extension to the SPIFFE, OAuth Token Exchange, and Rich Authorization Requests foundations this paper builds on [22], [15], [23] rather than a replacement for them. Fourth, the manual translation effort required to build the compliance control library in Section 4.4 suggests an opportunity for regulators and standards bodies to publish machine evaluable reference implementations of key requirements alongside the human readable text of a regulation; absent that, industry consortia such as the Cloud Security Alliance are a plausible near term venue for maintaining shared, peer reviewed control libraries that individual organizations could adopt rather than building from scratch, reducing the interpretive risk discussed in Section 7.2.

9. Conclusion and Recommendations

Agentic AI workloads have introduced a pattern of dynamic, delegated, machine initiated credential use that the identity and secrets management infrastructure of regulated industries was not built to govern, and the industry data reviewed in Section 2 show the gap widening rather than closing on its own. This paper presented the Agentic Secrets Governance Framework, which combines workload identity attestation, policy as code authorization, and a compliance control library that expresses HIPAA, PCI DSS, DORA, NIST AI RMF, and ISO 42001 obligations as machine evaluable policy, so that every credential decision is a security decision and a compliance decision evaluated together. A reference implementation, evaluated against healthcare eligibility and card payment authorization case studies, reduced the configured credential exposure window by more than four orders of magnitude relative to a static credential baseline while adding a combined median of 34.3 milliseconds to the request path, a cost that was not the dominant contributor to end to end response time in either workflow tested. The framework does not resolve every risk in the OWASP Top 10 for Agentic Applications, and it does not substitute for qualified legal and compliance review of the underlying control library, but it demonstrates that treating

compliance as executable policy rather than as retrospective documentation is a technically achievable and operationally meaningful step toward governing secrets in agentic AI workloads.

Based on these findings, the study makes three recommendations. Organizations in regulated industries should issue agent credentials through a policy decision point with task scoped time to live ceilings, link every authorization rule to a named regulatory provision, and assign explicit ownership of the resulting control library. Regulators and standards bodies should consider publishing machine evaluable reference controls alongside regulatory text and clarifying how existing provisions on unique identification and credential rotation apply to autonomous agents. Researchers should evaluate agent credential governance independently using public benchmarks and continue work on verifiable delegation and intent verification.

References

[1] CrewAI, "2026 State of Agentic AI Survey," Feb. 11, 2026. [Online]. Available: https://www.businesswire.com/news/home/20260211693427/en/

[2] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz, "Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection," in Proc. 16th ACM Workshop on Artificial Intelligence and Security (AISec), Copenhagen, Denmark, Nov. 2023, pp. 79-90, doi: 10.1145/3605764.3623985.

[3] Q. Zhan, Z. Liang, Z. Ying, and D. Kang, "InjecAgent: Benchmarking indirect prompt injections in tool-integrated large language model agents," in Findings of the Association for Computational Linguistics: ACL 2024, Bangkok, Thailand, Aug. 2024.

[4] E. Debenedetti, J. Zhang, M. Balunović, L. Beurer-Kellner, M. Fischer, and F. Tramèr, "AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents," in Advances in Neural Information Processing Systems 37 (NeurIPS 2024), Datasets and Benchmarks Track, 2024.

[5] M. Meli, M. R. McNiece, and B. Reaves, "How bad can it git? Characterizing secret leakage in public GitHub repositories," in Proc. Network and Distributed System Security Symposium (NDSS), San Diego, CA, USA, Feb. 2019, doi: 10.14722/ndss.2019.23418.

[6] A. Nabiullina and C. Winqwist, "The State of Secrets Sprawl 2026," GitGuardian, Mar. 17, 2026. [Online]. Available: https://www.gitguardian.com/state-of-secrets-sprawl-report-2026

[7] Cloud Security Alliance, "The Non-Human Identity Governance Vacuum: Agentic AI Governance," CSA Whitepaper, May 20, 2026. [Online]. Available: https://labs.cloudsecurityalliance.org/research/csa-whitepaper-nonhuman-identity-agentic-ai-governance-v1-cs/

[8] T. Otsuka, K. Toyoda, and A. Leung, "AI Identity: Standards, Gaps, and Research Directions for AI Agents," arXiv:2604.23280, Apr. 28, 2026.

[9] HashiCorp, "Advancing AI Agent Security in Vault," HashiCorp Blog, Jun. 23, 2026. [Online]. Available: https://www.hashicorp.com/en/blog/advancing-ai-agent-security-in-vault

[10] HashiCorp, "SPIFFE: Securing the Identity of Agentic AI and Non-Human Actors," HashiCorp Blog, 2026. [Online]. Available: https://www.hashicorp.com/en/blog/spiffe-securing-the-identity-of-agentic-ai-and-non-human-actors

[11] Y. Ruan, H. Dong, A. Wang, S. Pitis, Y. Zhou, J. Ba, Y. Dubois, C. J. Maddison, and T. Hashimoto, "Identifying the risks of LM agents with an LM-emulated sandbox," in Proc. 12th Int. Conf. Learning Representations (ICLR), Vienna, Austria, 2024.

[12] OWASP Foundation, "OWASP Top 10 for Agentic Applications 2026," Generative AI Security Project, Dec. 9, 2025. [Online]. Available: https://genai.owasp.org/resource/owasp-top-10-for-agentic-applications-for-2026/

[13] Cloud Security Alliance, "MITRE ATT&CK and ATLAS Agentic Gap Analysis: Techniques Unique to Autonomous Agent Control Planes," CSA Research Note, Mar. 27, 2026. [Online]. Available: https://labs.cloudsecurityalliance.org/agentic/csa-research-note-atlas-agentic-gap-analysis-20260327/

[14] S. K. Basak, L. Neil, B. Reaves, and L. Williams, "What are the practices for secret management in software artifacts?," in Proc. IEEE Secure Development Conference (SecDev), 2022, pp. 69-76.

[15] T. Lodderstedt, J. Richer, and B. Campbell, "OAuth 2.0 Rich Authorization Requests," RFC 9396, Internet Engineering Task Force, May 2023.

[16] HashiCorp, "Manage Vault policies with Sentinel," Vault Enterprise Documentation, 2026. [Online]. Available: https://developer.hashicorp.com/vault/docs/enterprise/sentinel

[17] HashiCorp, "SPIFFE auth method," Vault Documentation, 2026. [Online]. Available: https://developer.hashicorp.com/vault/docs/auth/spiffe

[18] SPIFFE Project, "SPIRE Concepts," Cloud Native Computing Foundation, 2026. [Online]. Available: https://spiffe.io/docs/latest/spire-about/spire-concepts/

[19] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero Trust Architecture," NIST Special Publication 800-207, National Institute of Standards and Technology, Aug. 2020, doi: 10.6028/NIST.SP.800-207.

[20] S. T. Avirneni, "Establishing Workload Identity for Zero Trust CI/CD: From Secrets to SPIFFE-Based Authentication," arXiv:2504.14760, Apr. 2025.

[21] J. W. Cutler et al., "Cedar: A new language for expressive, fast, safe, and analyzable authorization," Proc. ACM Program. Lang., vol. 8, no. OOPSLA1, Apr. 2024, doi: 10.1145/3649835.

[22] M. Jones, A. Nadalin, B. Campbell, J. Bradley, and C. Mortimore, "OAuth 2.0 Token Exchange," RFC 8693, Internet Engineering Task Force, Jan. 2020.

[23] T. South, S. Marro, T. Hardjono, R. Mahari, C. D. Whitney, D. Greenwood, A. Chan, and A. Pentland, "Authenticated delegation and authorized AI agents," arXiv:2501.09674, Jan. 2025.

[24] G. Syros, A. Suri, J. Ginesin, C. Nita-Rotaru, and A. Oprea, "SAGA: A security architecture for governing AI agentic systems," in Proc. Network and Distributed System Security Symposium (NDSS), San Diego, CA, USA, 2026; arXiv:2504.21034.

[25] Microsoft Security, "Shai-Hulud 2.0: Guidance for Detecting, Investigating, and Defending Against the Supply Chain Attack," Microsoft Security Blog, Dec. 9, 2025.

[26] Open Policy Agent, "OPA Documentation," Cloud Native Computing Foundation, 2026. [Online]. Available: https://www.openpolicyagent.org/docs

[27] Open Policy Agent, "Gatekeeper: Policy controller for Kubernetes" and "Conftest," project documentation, 2026. [Online]. Available: https://open-policy-agent.github.io/gatekeeper/ and https://www.conftest.dev

[28] M. Kellogg, M. Schäf, S. Tasiran, and M. D. Ernst, "Continuous compliance," in Proc. 35th IEEE/ACM Int. Conf. Automated Software Engineering (ASE), 2020, pp. 511-523, doi: 10.1145/3324884.3416593.

[29] M. A. Varma, "Compliance-as-Code for Continuous PCI DSS 4.0 Validation in Cloud-Native Financial Systems: Principles, Patterns, and Industrial Experience," International Journal of Intelligent Systems and Applications in Engineering, vol. 14, no. 1s, May 2026.

[30] U.S. Department of Health and Human Services, Office for Civil Rights, "Notice of Proposed Rulemaking: HIPAA Security Rule," Federal Register, Jan. 2025.

[31] PCI Security Standards Council, "Payment Card Industry Data Security Standard, Requirements and Testing Procedures, Version 4.0.1," 2024.

[32] European Parliament and Council of the European Union, "Regulation (EU) 2022/2554 on Digital Operational Resilience for the Financial Sector (DORA)," in force Jan. 17, 2025.

[33] National Institute of Standards and Technology, "Artificial Intelligence Risk Management Framework (AI RMF 1.0)" and "Generative Artificial Intelligence Profile (NIST AI 600-1)," NIST, 2023-2024.

[34] International Organization for Standardization, "ISO/IEC 42001:2023, Information Technology, Artificial Intelligence Management System," Dec. 2023.

[35] Google Cloud, "Workload Identity Federation," Cloud IAM Documentation, 2026. [Online]. Available: https://cloud.google.com/iam/docs/workload-identity-federation

[36] Amazon Web Services, "What is AWS Identity and Access Management Roles Anywhere?," IAM Roles Anywhere User Guide, 2026. [Online]. Available: https://docs.aws.amazon.com/rolesanywhere/latest/userguide/introduction.html

[37] Microsoft, "What are workload identities?," Microsoft Entra Workload ID Documentation, 2026. [Online]. Available: https://learn.microsoft.com/en-us/entra/workload-id/workload-identities-overview

[38] Microsoft, "Microsoft extends Zero Trust to secure the agentic workforce," Microsoft Security Blog, May 19, 2025. [Online]. Available: https://www.microsoft.com/en-us/security/blog/2025/05/19/microsoft-extends-zero-trust-to-secure-the-agentic-workforce/

2026 by the Authors. This Article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/)