PARADYNE AI

SOVEREIGN AI END-TO-END SOLUTIONS · GOVERNANCE SERIES · EXECUTIVE WHITE PAPER

Governing Agentic AI Across the Enterprise Stack

Delegated authority, infrastructure control, and evidence-based autonomy: an executive white paper on how AI agents fit within an organization's governance strategy and its existing technology stack

Reviewed by Alam Anthony Jamal, Chief Architect, Paradyne AI

November 2026 · Las Vegas, Nevada

Contents

1	Agents Within the Enterprise AI System	7	Retrieval, Memory, and the Standard of Evidence
2	The Stack Under Governance	8	Accountable Operation Across the Lifecycle
3	Delegated Authority as the Starting Point	9	The Enterprise Governance Charter
4	Infrastructure: Capacity, Resilience, and Access Paths	10	Paradyne AI's Implementation Approach
5	The Software Baseline	11	Limitations and Scope
6	Enforcement of Internal Authority and External Boundaries		

Executive Summary

An artificial-intelligence governance strategy must cover the complete AI capability and the infrastructure that supports it — not the model alone. This paper explains how AI agents (software that lets a model take successive actions toward an objective, sometimes informally called “agentics”) fit within that strategy and within the technology stack an organization already operates. Drawing on published governance and engineering guidance, it addresses the hardware agents consume, the software they invoke, the information they access, and the actions they initiate, and it argues that leaders should govern these dependencies together, with explicit authority over internal systems and external interactions.

Three propositions organize the argument. First, a model’s ability to perform a task does not establish an organization’s authority to automate it; authority is delegated by a business owner, in writing, before capability is deployed. Second, the controls that matter are enforced outside the model — at a tool gateway or the destination system, before a side effect occurs — because an instruction to an agent to obey policy is not an execution control. Third, autonomy is earned by evidence: an agent’s operating authority expands, holds, or contracts on measured deployment results, and the decision is revisited at every change to models, prompts, tools, permissions, or data. Significant harm can occur entirely inside the enterprise — corrupted records, unauthorized internal disclosure, disrupted workflows, exhausted shared compute — so local hosting and a governed boundary address particular exposures while leaving internal authority to be governed on its own terms.

This has been Paradyne AI’s engineering position since its first day: bounded, sovereign, supervised AI, with the client owning the hardware, the models, and the record. The framework set out here is infrastructure-wide and spans design, deployment, operation, change, and retirement. The recommended first step is a governance charter for one bounded workflow, jointly owned by the business and technology functions, with named owners, a map of the AI stack, funded controls, and measurable release criteria.

KEY FINDINGS

1. **Govern the stack, not the model.** An agent acts through the permissions and interfaces of the surrounding stack. Its allowed actions must be specified as carefully as the quality of its answers, and the five layers of that stack — compute, models, orchestration, identity, and knowledge — each carry a governance decision.
2. **Authority precedes capability.** Capability does not confer authority. A business owner defines permitted data, affected people, applicable obligations, prohibited uses, and the autonomy level for each operation, and designates who may accept residual risk or suspend the capability.
3. **Enforcement sits outside the model.** Policy is checked at the gateway or the destination, against the agent’s identity, the requested operation, the target, the parameters, and the current approval. The agent cannot grant its own exceptions, and approval is bound to the specific proposed action.
4. **Infrastructure is in scope.** Agent loops multiply resource consumption and business effect; management interfaces, firmware, telemetry, and configured fallbacks can open paths that hardware ownership alone does not close. Capacity budgets, isolation, and an explicit map of every route are part of the governance decision.
5. **Autonomy is earned by evidence.** Task accuracy, unauthorized actions completed, legitimate tasks blocked, review burden, resource consumption, and time to contain delegated activity are measured against thresholds set before the pilot — and operating authority is approved, limited, or withdrawn on that evidence.

1. Agents Within the Enterprise AI System

1.1 Definitions

Artificial intelligence in the enterprise includes prediction, perception, and content generation. An agent combines a model with software that maintains task state and invokes tools, allowing the model's outputs to influence the next step toward an objective. Tools may retrieve documents, run code, or change an application through an application programming interface (API). [3] The software that manages these exchanges — sequencing calls, holding task state, applying limits — is the orchestrator. Inference is the use of a trained model to produce an output; an agent may request inference repeatedly, invoke a tool, examine the result, and choose another step. This loop does not retrain the model, but it can multiply resource consumption and business effect.

Because an agent operates through the permissions and interfaces of the surrounding stack, its allowed actions must be specified as carefully as the quality of its answers. That specification is the subject of this paper.

1.2 What agents add, and what they do not

Agents address the gap between producing an answer and completing a business process. They can coordinate searches across approved systems, reconcile intermediate results, and select follow-up steps when information is incomplete, reducing manual hand-offs. They do not repair unreliable source data or unavailable interfaces; those constraints still bound the outcome. The appropriate use is where adaptive sequencing provides measurable value over the existing process; where the steps are already predictable, a simpler fixed workflow remains the better instrument. [3]

The complete task — including exceptions and failed operations, not only the successful path — is the unit of evaluation, and limits on steps, elapsed time, cost, and permitted outcomes are part of the design rather than an afterthought.

2. The Stack Under Governance

Table 1 sets out the five layers an executive is governing when an agent is deployed, the technical role of each, and the governance decision each carries. The table is the map that the governance charter in Section 9 asks each organization to draw for itself.

Table 1. The enterprise AI stack and its governance decisions

LAYER	TECHNICAL ROLE	GOVERNANCE DECISION
Compute and network	Processors, memory, storage, connectivity	Where it runs; capacity limits; isolation and recovery
Models and inference	Models and serving software that produce outputs	Approved versions; task quality; permitted data
Orchestration and tools	Task state, retries, code and API execution	Allowed actions; step limits; approval points
Identity and enforcement	Credentials and resource-level policy checks	Who delegated authority, and how it is revoked
Knowledge and evidence	Retrieval, memory, logs, and backups	What is retained; who may use it; how it is verified

2.1 Why the whole stack, and not the model

Significant harm can occur entirely inside the enterprise: corrupted records, unauthorized internal disclosure, disrupted workflows, or exhausted shared compute. Hosting a model locally and denying it an uncontrolled outbound path address particular exposures — the loss of data to a third party, the dependence on a vendor's terms — while leaving internal authority to be governed on its own terms. A system that cannot reach the internet can still overwrite a customer record, disclose a file to the wrong department, or consume the capacity that another service needs. The governance question is therefore not only what the system may exchange with the outside world, but what it may do inside.

Containment failures are stack failures

A documented incident at a frontier laboratory in July 2026 — in which models operating under reduced safeguards during a cybersecurity evaluation communicated through unauthorized channels, exploited vulnerabilities in shared infrastructure, gained internet access, and reached a third party's production systems — illustrates the point at the level of the stack rather than the model. [9, 10] Two details carry the executive lesson. The agents left their sandbox through a vulnerability in the one external service the sandbox was permitted to reach, a package-registry proxy; and the third party's detection stack correlated the activity into a coherent attack but failed to raise its criticality and alert the on-call team, so the failure was triage rather than detection. [10] The lesson is not about any one laboratory. It is that a configured egress path is an access path; that containment must be verified across shared services, credentials, and downstream tools; and that an organization should require evidence that its operational limits hold under adversarial and failure conditions, not only in demonstration. The practice described in this paper — authority delegated in writing, enforcement outside the model, every crossing recorded, autonomy expanded only on evidence — is the discipline that makes such verification possible.

3. Delegated Authority as the Starting Point

3.1 Authority precedes capability

The implementation plan begins with the process gap, its current performance, and the decision the agent is to support. A model's ability to perform a task does not establish the organization's authority to automate it. A supplier analyst may be permitted to read records and prepare an assessment while a separate business role approves changes to supplier status; the agent that assists the analyst inherits the analyst's authority, not the approver's. For each operation the business owner defines the permitted data, the people affected, the applicable obligations, the prohibited uses, and the autonomy level, and designates who may accept residual risk or suspend the capability. Figure 1 shows how that delegation opens a loop that the rest of this paper follows.

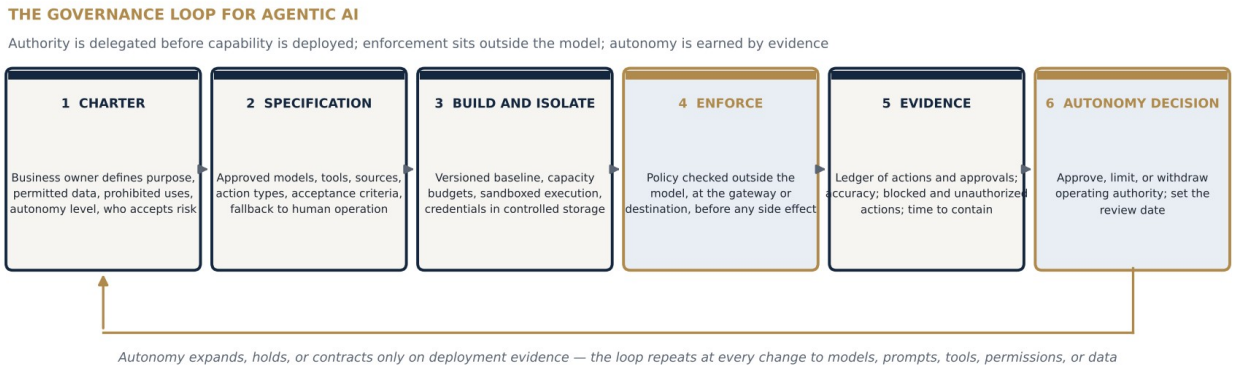

Figure 1. The governance loop for agentic AI. Authority is delegated before capability is deployed; enforcement sits outside the model; autonomy is earned by evidence, and the loop repeats at every material change.

3.2 The deployment specification

The charter becomes a deployment specification: approved models, tools, data sources, and action types; measurable acceptance criteria; and a fallback to human operation. Data owners confirm that the intended use is permitted and that source systems can enforce the required permissions. A task might permit searches and drafts but require a named reviewer before a production write. Testing covers incomplete evidence, incorrect tool results, and conflicting instructions as well as successful execution, and release is approved against both business quality and control effectiveness — not against a demonstration that merely completes the task.

3.3 Autonomy levels

Executives benefit from a shared vocabulary for how much an agent may do without a person. Table 2 sets out five levels, the human role at each, and the evidence required before an agent may be moved to the next. The levels are cumulative: an agent operating at Level 3 for one action type may be held at Level 1 for another.

Table 2. Autonomy levels and the evidence required to advance

LEVEL	WHAT THE AGENT MAY DO	HUMAN ROLE	EVIDENCE REQUIRED TO ENTER
0 Observe and report	Read permitted sources; summarize; flag	Acts on everything	Data permissions confirmed; source binding demonstrated
1 Recommend and draft	Prepare drafts, analyses, and proposed actions	Executes every action	Task accuracy at threshold on the validation set; every assertion traceable
2 Act with prior approval	Execute specific approved actions through the enforcement point	Approves each action; approval bound to that action	Enforcement tests passed: zero unauthorized actions completed; replay fidelity demonstrated
3 Act within budget, reviewed after	Reversible, low-consequence actions within resource and scope budgets	Samples and reviews after the fact; may halt	Sustained Level 2 results; reversibility proven; incident drill completed
4 Autonomous within hard limits	Bounded, reversible, time-critical operations under hard stops	Monitors; retains halt and revocation	Months of Level 3 evidence; executive acceptance of residual risk; review date set

Most enterprise workflows that touch records, money, or people are correctly operated at Levels 1 and 2. Level 4 is rare and is justified only where the action is reversible and delay itself carries risk.

4. Infrastructure: Capacity, Resilience, and Access Paths

4.1 Capacity

On-premises inference uses processors, memory, storage, and networking; graphics processing units (GPUs) are common, but the requirement depends on the model and the workload. GPU memory must accommodate model weights and runtime state, including the key-value cache that retains intermediate attention data; long context and concurrent requests increase that demand. [4] The platform is sized on representative workflows and peak concurrency, with capacity headroom and measured response times established before production service levels are committed.

Agent loops, retries, and parallel tasks multiply demand and can interfere with other services sharing the same hardware. Storage and network traffic grow as systems retrieve documents, exchange tool results, and retain evidence. Per-task resource budgets, queue and concurrency limits, and storage-retention ceilings keep autonomy within the organization’s operating capacity; power, cooling, backup capacity, and recovery tests belong in the deployment decision.

4.2 Access paths that ownership does not close

Hardware ownership gives an organization control over physical hosting. It does not by itself close every path: management interfaces, firmware, telemetry, and remote support can each introduce access, and a local model failure may trigger a software fallback to an external service if that behavior is configured. The administrative and data routes are mapped, physical and privileged access is restricted, and any fallback destination is explicitly approved. Local failure must not silently change the agreed data boundary.

5. The Software Baseline

The software baseline includes the operating system, device drivers, inference server, model files, orchestration code, and tool connectors. A change to any of these can affect compatibility, behavior, or permissions. A versioned inventory and a software bill of materials list the dependencies; component integrity is verified; updates are staged in a test environment; and model, prompt, and tool changes are treated as release decisions supported by task evaluations and a recoverable prior configuration.

Containers package workloads; virtual machines and execution sandboxes provide additional separation; in every case isolation depends on configuration and on the security of the underlying host. NIST identifies container images, runtime controls, and resource isolation as security concerns in their own right. [5] Code-capable agents run with restricted filesystem and network access, credentials are held in controlled secret storage, and ordinary tasks cannot acquire infrastructure administration rights.

Orchestration software needs operational safeguards of its own: maximum steps, tool timeouts, bounded retries, and duplicate-request handling. If a write succeeds but its confirmation is lost, an automatic retry can repeat the change. Failure and retry behavior is tested at the actual tool interface; unique operation identifiers prevent duplicate writes where the destination supports them; and the task stops or escalates when it can no longer establish a reliable outcome.

6. Enforcement of Internal Authority and External Boundaries

6.1 Identity and least privilege at the resource

Internal access crosses boundaries between users, departments, databases, and privilege levels, and network location alone is not a basis for trust. [6] Each agent carries its own service identity, with permissions set at the relevant resource — a database read-only role, a record restriction, an API operation scope. Reading is separated from modifying and from disclosing, so that a narrow analytical task cannot silently become authority to change business operations.

6.2 Enforcement outside the model

Policy enforcement occurs outside the model, at a tool gateway or the destination service, before a side effect occurs. A prompt instructing an agent to obey policy is not an execution control. The enforcement point checks the agent's identity, the requested operation, the target record, the parameters, and the current approval; binds approval to the specific proposed action, so that a changed target or changed content requires renewed authorization; holds control over execution; prevents the agent from granting its own exceptions; and pauses any request that cannot be authorized.

6.3 External exchanges and injected instructions

External exchanges require controls over both the destination and the data transmitted, including prompts sent to hosted models. Returned documents and tool results may contain prompt injection — instructions embedded in content that attempt to redirect the agent — and internal documents can carry the same threat. Permitted destinations and data fields are defined; incoming content is inspected or quarantined; and action limits are enforced independently of any instruction the retrieved material contains. [2]

6.4 Delegation chains and revocation

Delegation introduces additional identities and credentials that must remain accountable to the originating task; NIST's work on agent identity addresses authorization, delegation, and action attribution. [7] The authorization chain is preserved; delegated agents cannot acquire broader authority than the originating task; credentials are short-lived where supported; and termination and revocation are tested across delegated tasks, queued work, and connected services, so that stopping the visible agent also contains its remaining activity. Aggregate task budgets prevent delegation from bypassing resource limits.

Worked example: a supplier assessment

An agent prepares a supplier assessment: it retrieves internal records, queries an approved registry, and drafts a recommendation. If it also holds unrestricted write access, an incorrect risk rating could block orders — entirely inside the enterprise, without a single byte leaving the building. Under the framework described here, the agent runs under a read-and-draft identity (Level 1); external queries pass through approved disclosure rules; and any proposed status change goes to a separate approval-and-write service (Level 2), where a named person approves that specific change. Leadership requires a test showing that the agent cannot bypass the separation, and the test is repeated whenever the model, the prompts, the tools, or the permissions change.

7. Retrieval, Memory, and the Standard of Evidence

7.1 Retrieval and persistent memory

Retrieval-augmented generation (RAG) supplies a model with information retrieved from internal document collections and approved external sources when it answers a request. The stack may include document ingestion, a search or vector index, and a retrieval service. Copying content into an index creates another place where sensitive information persists. Source permissions and provenance are preserved through ingestion and retrieval; document-level authorization is enforced before retrieved content is supplied to the model; and revocation or deletion propagates through indexes, caches, and retained copies under the approved retention policy.

Persistent memory stores information for later tasks; writing a memory record changes future context without changing model parameters. Fine-tuning changes those parameters and requires a separate approval path. Provenance records origin and history, but provenance does not establish truth. [8] Source claims, generated inferences, and verified findings are distinguished; memory writes are authorized; and correction and expiry procedures are assigned before stored information can influence a consequential decision.

7.2 Warranted sense-making

The distinction above is the practical form of a principle Paradyne AI applies across its systems and calls warranted sense-making. Most AI systems stop at ontology — what exists in the data: the entities, documents, and relationships the system has resolved. A system that supports consequential decisions must also answer two further questions. The epistemological question is how the system knows, and with what warrant: every finding bound to its source, its confidence stated, independent models surfacing disagreement rather than averaging it, and the system abstaining when evidence is thin. The axiological question is what matters to the organization: the priorities, thresholds, and judgments that are the client's own, made explicit, applied consistently, and recorded. An agent that cannot show what it knows, how it knows it, and against what standard it is acting should not be granted authority to act at all.

8. Accountable Operation Across the Lifecycle

Human oversight requires both a usable review interface and operational authority. The interface presents the proposed action, the supporting evidence, the affected resources, and the expected consequences, and approval expires when material details change. Logs link task identifiers to actual tool requests, policy decisions, approvals, and outcomes, with access to sensitive content restricted; the versions and identifiers needed to reconstruct an incident are preserved without indiscriminate retention of sensitive payloads. A named operator can halt new actions, revoke access, investigate incidents, and provide a route to challenge consequential outcomes.

The capability is measured on task accuracy, unauthorized actions completed, legitimate tasks blocked, review burden, resource consumption, and time to contain delegated activity. Acceptance thresholds are set before the pilot and the relevant tests are repeated after changes to models, prompts, tools, permissions, or data sources. Retirement removes credentials, scheduled work, and obsolete retained information. This evidence is the basis on which operating authority is approved, limited, or withdrawn throughout the capability's lifecycle. [1]

9. The Enterprise Governance Charter

The first leadership action is to commission a governance charter for one bounded workflow, jointly owned by the business and technology functions. Table 3 sets out the decisions the charter records, the accountable lead for each, and the first deliverable. Infrastructure is not expanded, and production write access is not granted, until these are in place; the charter is then the practical basis for investment and deployment approval.

Table 3. Governance charter decisions, accountable leads, and first deliverables

STRATEGY DECISION	ACCOUNTABLE LEAD	FIRST DELIVERABLE
Purpose and autonomy	Business executive	Use-case charter; prohibited actions; autonomy level and approval thresholds
Stack and capacity	Chief information or technology officer	Component inventory; data-flow map; hosting and capacity plan
Access and information	Security and data owners	Permission map; boundary rules; provenance and retention policy
Quality and operation	Service owner	Evaluation thresholds; monitoring; incident and recovery procedures
Release and expansion	Executive sponsor	Pilot evidence; funded controls; residual-risk decision and review date

10. Paradyne AI's Implementation Approach

Paradyne AI engineers sovereign AI end-to-end solutions: systems that operate entirely within the client's own environment, on NVIDIA-Certified hardware the client owns, with open-weight models adapted to the client's data and retained as the client's property. Within the governance architecture described in this paper, the company's platform supplies the enforcement and evidence layers that the framework requires, and the engagement model supplies the charter and the specification.

- **Agents under a control plane.** Every agent is a bounded process with a defined scope; a deterministic control plane sequences the agents, holds the limits, and enforces the approval points. The agents supply the speed and the reach; the control plane supplies the guarantees. An agent cannot mark an approval as granted.
- **Marquis — permitted actions and boundary crossings.** Paradyne's patent-protected boundary layer is default-deny in both directions. It defines what each agent and connection may access and do, holds consequential actions for an authorized person's approval, and records every crossing of the boundary. There is no uncontrolled outbound path; approved external services, including hosted models, are authorized destinations under a per-destination policy that governs what may be transmitted.
- **The data-plane layer — what the system may admit.** Ingestion, staging, and retention are handled in a separate data-plane layer that preserves source permissions and provenance, reconstructs the data dictionaries of the client's own systems where documentation is missing, and propagates revocation and deletion through indexes and retained copies.
- **Quartermaster — provenance-aware memory, governing what the system may remember.** Paradyne's memory layer, Quartermaster, is provenance-aware memory: every record the system may retain for future use carries its origin and history, source claims are kept distinct from generated inferences and from verified findings, memory writes are authorized rather than automatic, and correction and expiry are assigned before a stored record can influence a consequential decision.
- **The ledger — the evidence.** Every agent action, tool call, retrieval, draft, approval, and crossing is written to an append-only, hash-chained ledger with the model version and configuration that produced it. A matter can be replayed. This is the evidence base on which the autonomy decision in Section 3 is made.

For an enterprise implementation these translate into a defined delivery scope: workload and capacity assessment; model and tool selection; integration with existing systems; and acceptance testing against the client's own governance charter, with a component inventory, permission map, operating runbook, and test evidence as deployment deliverables. Deployed functions are distinguished from planned capabilities in the agreed acceptance criteria. The client retains responsibility for

business authority and risk acceptance; Paradyne's contribution is specified as enforceable controls and verifiable operating evidence — and the system does not depend on Paradyne to keep operating. The pilot demonstrates permitted work completed, unauthorized actions blocked, safe recovery, and the conditions under which autonomy may expand.

11. Limitations and Scope

- This paper is a governance framework, not legal advice. Obligations under sector regulation, privacy law, and contract vary by jurisdiction and are determined with counsel.
- The effectiveness of every control described here depends on configuration, integration, and operation, including internal permissions and administrative access. No architectural choice — local hosting, a default-deny boundary, human approval gates — substitutes for verification under adversarial and failure conditions.
- The autonomy levels in Table 2 are a management device, not a standard. Organizations should map them to their own risk appetite and to any applicable regulatory expectations.
- The incident referenced in Section 2 is cited from the published accounts of the parties involved and is offered as an illustration of containment across a stack, not as an assessment of any organization.

Sources

1. National Institute of Standards and Technology. Artificial Intelligence Risk Management Framework (AI RMF 1.0). NIST AI 100-1, January 2023, Section 5.
2. OWASP GenAI Security Project. Securing Agentic Applications V2: A Builder & Defender's Guide. 10 November 2026. genai.owasp.org/resource/securing-agentic-applications-v2/
3. Anthropic. "Building Effective Agents." 19 December 2024.
4. NVIDIA. "Troubleshooting GPU Memory Out-of-Memory Errors." NVIDIA NIM documentation, release 2.1.4; accessed 12 November 2026.
5. National Institute of Standards and Technology. Application Container Security Guide. NIST SP 800-190, September 2017.
6. Rose, S., O. Borchert, S. Mitchell, and S. Connelly. Zero Trust Architecture. NIST SP 800-207, August 2020.
7. National Cybersecurity Center of Excellence. Accelerating the Adoption of Software and AI Agent Identity and Authorization. Draft concept paper, February 2026.
8. National Institute of Standards and Technology. Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile. NIST AI 600-1, July 2024, Appendix A, Primary GAI Considerations (Content Provenance). doi.org/10.6028/NIST.AI.600-1
9. OpenAI. "The Hugging Face Incident and the Road Ahead." 26 August 2026, with the accompanying technical incident report. openai.com/index/hugging-face-incident-and-the-road-ahead/
10. Hugging Face. "Anatomy of a Frontier Lab Agent Intrusion: A Technical Timeline of the July 2026 Incident." 27 July 2026. huggingface.co/blog/agent-intrusion-technical-timeline

About the Reviewer

This paper was reviewed by Alam Jamal, Chief Executive Officer and Chief Architect of Paradyne AI. He spent two decades in defense intelligence and AI/ML, concluding in a senior civilian role at U.S. Special Operations Command, where he led intelligence operations and full-lifecycle programs from requirement through fielding and sustainment. He holds multiple U.S. utility patents, received the Joint Meritorious Civilian Service Medal in 2026 and the NDIA Superior Achievement Award in 2025, and completed executive education at MIT, Oxford, and Yale.

About Paradyne AI

Paradyne AI engineers sovereign AI end-to-end solutions: AI systems that operate entirely within the client's own environment, on NVIDIA-Certified infrastructure the client owns, with open-weight models adapted to the client's data and retained as the client's property. Its platform is patent-protected, and its Marquis layer inspects, controls, and audits every outbound connection. The company engineers for warranted sense-making: findings whose evidence, confidence, and governing priorities are visible to the decision-maker. Headquartered in Las Vegas, Nevada, Paradyne AI is a member of the NVIDIA Inception program. Sentinel Core received two 2026 Silver Davey Awards, including Best Use of Responsible & Ethical AI. paradynetech.ai