Controls for both surfaces

Two surfaces arrive with LLMs: attackers iterating at machine speed against your perimeter, and agents inside your environment processing untrusted text next to privileged tools. A working set of controls for each, and the principles they turn out to share.

Matthew J. Harmon

2026-09-13

Attacking the external surface (LLM-armed attackers)

LLM-assisted tools drastically reduce the time, cost, and effort required to execute traditional attacks. Machine-speed reconnaissance and payload iteration change what "good enough" defense looks like.

- **Eliminate exposed services:** Deploy continuous external attack-surface discovery with explicit asset ownership to identify forgotten staging APIs, legacy portals, and management interfaces.
- **Patch by exposure, not CVSS alone:** Shrink patch windows for internet-facing systems, identity infrastructure, and vulnerabilities with public exploits.
- **Design for credential compromise:** Implement phishing-resistant authentication (FIDO2/WebAuthn, passkeys) alongside conditional access and short-lived privileged sessions to ensure passwords alone are never sufficient.
- **Detect operational velocity:** Instrument behavioral detections to flag anomalous speed—such as rapid authentication attempts, service enumeration, or sudden API expansion—rather than relying solely on static signatures.
- **Deploy deception as telemetry:** Place honeytokens, canary credentials, and decoy records in high-value paths. Any interaction with these resources serves as a high-fidelity alert signal.
- **Apply targeted friction:** Protect unauthenticated endpoints (login, recovery, registration) using rate limits, quotas, progressive delays, and behavioral challenges to stop low-cost, automated attacks.

Securing the internal surface (LLMs inside your environment)

When AI models and agents operate inside your network, treat them as untrusted software principals capable of being manipulated. Their risk increases with the combination of input, authority, tools, and autonomy.

- **Enforce strict minimum authority:** Assign dedicated, scoped, read-only, and short-lived identities to each agent. Avoid inheriting host permissions or combining roles across workflows.

- **Treat retrieved content as untrusted:** Assume any data processed by an agent—emails, documents, search results, or API responses—can contain indirect prompt injection. Never let raw text define operational authority.
- **Decouple reasoning from authorization:** The model suggests actions; an external, deterministic policy engine enforces permissions using verified contextual facts (identity, target, action type, destination).
- **Restrict outbound traffic (egress):** Block direct egress to internal metadata services, route external traffic through monitored proxy checkpoints, and separate internal and external network paths.
- **Log actions over context:** Retain structured, integrity-protected logs capturing the agent identity, invoked tool, parameters, target resource, and authorization decision. Focus forensic telemetry on actions taken rather than chat context.
- **Require independent approval for high-impact actions:** Force human review or multi-party policy checks before performing sensitive operations (IAM changes, data deletion, financial transactions, code deployment).
- **Sandbox runtime environments:** Run agent tasks inside isolated, short-lived containers or microVMs with strict memory, process, filesystem, and network constraints. Destroy the environment upon task completion.
- **Manage agents as supply chains:** Track and version system prompts, MCP servers, plugins, orchestration frameworks, and vector stores with the same rigor applied to executable code.

Key takeaway

LLMs allow external attackers to automate rapid iteration and allow internal agents to process untrusted data near privileged tools. Defensive resilience relies on familiar principles—**strict identity boundaries, least privilege, explicit policy enforcement, and complete action logging**—executed at machine speed.