01AI LTD

WHITEPAPER

The Agentic Blast Radius

Why Your AI Agents Are Your Next Insider Threat

Five control gaps that turn autonomous AI from a productivity multiplier into a breach multiplier.

April 2026
01AI LTD · Dublin, Ireland
www.01ltd.com

Abstract

Enterprises spent the last decade hardening human identity. Single sign-on replaced password sprawl, multi-factor authentication became table stakes, role-based access control matured into fine-grained authorisation, and security operations centres developed the telemetry and response playbooks to detect when a human credential was misused. The result was not perfect security, but it was a credible defensive posture against the insider threat.

Agentic AI is quietly undoing that work. Every control gap the industry has closed for human identity has reopened at machine speed, without intent, and with credentials most organisations have never inventoried. The 2026 attack surface is not the model (models have become commodity components) but what the model is allowed to do, who it is allowed to do it as, and how quickly it can do it before anyone notices.

Gartner projects that 33% of enterprise applications will embed agentic AI by 2028, up from less than 1% in 2024. IBM's 2025 Cost of a Data Breach Report found that breaches involving non-human identities now carry a higher average cost than those involving human credentials. These two trajectories are converging, and most enterprise security programmes are not prepared for the collision.

This paper identifies five control gaps that turn autonomous AI deployments from productivity multipliers into breach multipliers. Each gap maps to a control enterprises already possess for human users. The defensive work is not inventing new disciplines. It is extending existing ones (identity management, least privilege, data-loss prevention, retention, security operations) to a class of actor that the original designs did not anticipate.

The Shift No One Announced

In 2023, "AI in the enterprise" meant a chatbot. A user typed a prompt, a model responded, the interaction ended. The threat model was narrow: what comes out of the model, what goes into it. Prompt injection was a research curiosity. Data leakage was a training-time concern. Governance was about acceptable use policies.

By 2026, that framing is obsolete. The production pattern is an agent: a system that takes goals, decomposes them into sub-tasks, selects tools, makes API calls, writes to databases, sends emails, triggers workflows, and closes the loop by acting in the world on behalf of the user or the organisation. The agent pattern did not arrive through a single announcement. It accumulated through a series of incremental capabilities (function calling, tool use, the Model Context Protocol, multi-step reasoning) until one day the phrase "the AI did it" stopped being metaphorical.

Three characteristics of agentic AI matter for security. First, agents take actions, not just produce text. An agent that drafts an email is a productivity tool. An agent that sends one is an actor in your

systems. Second, agents operate continuously and at machine speed. A human insider compromised for a week might exfiltrate a few gigabytes; a compromised agent can do the same in minutes. Third, agents chain actions across tools, which means the blast radius of a single compromise is bounded not by the agent's direct permissions, but by the transitive closure of every system its tools can reach.

None of this maps cleanly onto the threat models that security teams developed for chatbots, for human users, or for conventional service accounts. The gaps below are the specific places where the mismatch has already produced incidents, and where we expect the majority of agentic AI breaches over the next eighteen months will be located.

1. Gap One: Identity Sprawl

The bot did it

The first gap is the one that makes every subsequent gap harder to investigate. Most agentic deployments in production today authenticate using a shared service account or a long-lived API key. There is no per-agent identity, no per-action attribution, and no revocation path short of rotating the shared credential and breaking every agent that depends on it.

The pattern emerges innocently. An engineering team builds an agent that needs to read from the CRM. They provision a service account called something like `ai-agent-prod`, grant it the necessary permissions, issue an API key, and move on. A second team builds a second agent for a different purpose. Rather than provisioning a new identity, they reuse the first one: the permissions are already approved, the key is already in the secrets manager, the path of least resistance is obvious. Six months later, a dozen agents, owned by four teams, share a single identity. When something goes wrong (an unauthorised data pull, a deleted record, a suspicious API call pattern) the audit trail collapses to "the bot did it."

This is the non-human identity problem, and it is the foundational failure of agentic AI security. Every control that depends on knowing who did what (authorisation, audit, anomaly detection, incident response) degrades to the extent that identity is shared across agents, workloads, and teams. The investigative question "which agent, running which task, on behalf of which user, took this action" is unanswerable in most current deployments.

IBM's 2025 breach data quantifies the consequence. Incidents involving non-human identities (service accounts, API keys, machine credentials, and now agent identities) now average a higher cost per incident than those involving human credentials, reversing a gap that held for most of the last decade. The driver is investigation time: without per-action attribution, mean time to identify stretches from days into weeks, and the downstream cleanup expands accordingly.

The human-world parallel is precise. An organisation that issued a single shared login to ten employees, with no per-person audit trail, would be in breach of basic identity hygiene and would likely fail its next compliance review. The same organisation has quietly replicated the anti-pattern

for its agents, because the agents are code, and code is a “system” to which those disciplines have not historically applied.

The corrective

Every agent deployed into production needs its own identity. Every action taken by an agent needs to be attributable to that identity. Every agent identity needs a documented owner, a purpose, an expiry, and a revocation procedure. The emerging category of non-human identity management tooling is beginning to make this tractable; the cultural shift inside engineering teams to treat agent identities with the same discipline as human identities has not caught up.

2. Gap Two: Permission Creep by Design

The second gap is about authorisation, and it is worse in agentic deployments than it ever was for human users, because it is designed in rather than accumulated.

When an organisation provisions a human user, the default posture is restrictive. The user gets the minimum permissions needed for their role, with additional access requested and approved on demand. Permission creep over time is a well-documented problem, but it creeps upward from a low baseline.

When an organisation provisions an agent, the default posture is permissive. The agent is scoped for the broadest task it might ever be asked to perform, not the narrowest task it is doing right now. A research agent that occasionally needs to update a record is given continuous write access to the CRM. A support agent that might need to reference a knowledge base is given read access to the entire document store. A coding agent that might touch a production repository is given commit rights. The reasoning is always the same: we do not know in advance what the user will ask, so we cannot scope the agent’s permissions to the specific task. The agent’s authority is the union of every task in its possible scope.

This is permission creep as architecture. And it is uniquely dangerous in agentic systems because of the interaction with the next gap: any agent with broad permissions is one successful prompt injection away from exercising those permissions against the organisation’s interests. A research agent that can write to the CRM can be induced to exfiltrate it, modify it, or poison it. A support agent with broad document-store access can be induced to leak the document it should never have been able to see. A coding agent with commit rights is a supply-chain attack surface that the organisation provisioned itself.

The OWASP Top 10 for LLM Applications calls this category “excessive agency,” and in the 2025 revision it was elevated to a first-tier risk precisely because the gap between public agent demos and production-ready architecture turned out to be wider than the 2023 list anticipated. Public demos routinely grant agents permissions that would be indefensible in production. The field-level reality is that most mature agent deployments now operate under substantially more constrained authority

than the demos, and most new deployments repeat the permissive pattern anyway, because the demo is the reference implementation the team starts from.

The corrective

An agent should have a role. The role should be scoped to a specific task or a narrow set of tasks. Permissions should be the minimum required for that role, not the maximum the agent might hypothetically need. Actions that exceed the role's authority should require escalation, ideally to a human, certainly to a separately-authorised path. The engineering discipline this requires is more intrusive than most teams expect. It means splitting monolithic agents into narrower agents with narrower scopes. It means building escalation paths rather than just-in-case permissions. It means accepting that some tasks the agent could theoretically perform will require a human in the loop. Every one of these decisions reduces the agent's apparent capability and increases its actual safety. The instinct to design agents as maximally capable is the instinct that creates the blast radius.

3. Gap Three: The Tool-Use Trust Boundary

The third gap is the one that most directly concerns practitioners watching the agentic ecosystem mature in real time. Every tool an agent can use (every MCP server, every plugin, every function call, every API integration) is a new trust edge in the organisation's security graph. Most of them are added without security review, because the framing is "it's just a tool the agent uses," not "it's a new path through which external data can influence our systems."

This framing is wrong, and the wrongness compounds.

The mechanism that makes tool use dangerous is indirect prompt injection. When an agent reads a document, fetches a web page, processes an email, or ingests the output of any tool that returns attacker-influenced content, the content of that data is interpreted by the model as instructions. A malicious document can instruct the agent to exfiltrate data. A poisoned web page can instruct the agent to take an action on behalf of the attacker. A compromised MCP server can inject instructions into every agent that connects to it. The attack surface is not the model; it is every source of input the model will eventually read.

This is not a theoretical concern. Indirect prompt injection has produced real incidents in real production systems over the last eighteen months, across every major agent framework and every class of tool integration. The defensive community has converged on the conclusion that prompt injection cannot be reliably prevented at the model layer, which means the defensive work has to happen at the architectural layer: treat every tool as a trust boundary, validate inputs and outputs across that boundary, and constrain what the agent can do with the results.

The Model Context Protocol, which has become the dominant standard for agent-tool integration over the last year, has made this problem simultaneously more tractable and more urgent. More tractable because MCP servers are discrete, inventoriable, and individually auditable. More urgent because the ease of adding MCP servers means that most enterprise agent deployments now have

dozens of them connected, with the security posture of the weakest determining the security posture of the whole.

The most common failure pattern we see in the field is that teams treat MCP servers and tool integrations as internal infrastructure that does not require the same review as external integrations. A team adds an MCP server that reads from Jira. Another adds one that reads from Confluence. A third adds one that reads from the company wiki. Each, individually, is unremarkable. Collectively, they form an attack surface where any user who can post to any of those systems can inject instructions into every agent that reads from any of them. The transitive compromise path is usually invisible to the teams that built each individual integration.

The corrective

Build a tool-use gateway: a single point through which all agent tool calls pass, where authorisation is enforced, where outputs are validated, where suspicious patterns are logged, and where the organisation can reason centrally about what its agents are allowed to do and what they are currently doing. This is the agent equivalent of the API gateway that the industry took a decade to standardise for web services, and it is where the most consequential security work on agentic AI will happen over the next two years.

4. Gap Four: Memory as a Data Leak

The fourth gap concerns persistence. Agents remember. They accumulate conversation history, write to vector stores, maintain long-term memory across sessions, and cache retrieved content for reuse. Each of these mechanisms is a data store, in the regulatory sense of that term, and most of them have been deployed without the data governance that the organisation's other data stores are subject to.

The failure modes are specific.

Conversation histories capture sensitive data entered by users: personal information, confidential plans, privileged communications, authentication secrets mistakenly pasted into a prompt. This data then persists in the conversation store, often outside the classification zone that should have applied to it. If a user enters client data into an agent conversation, the conversation store now contains client data, regardless of whether the conversation store was designed to hold it.

Vector stores (the embedding databases that power retrieval-augmented generation) hold representations of the documents they have ingested. Those representations are not the original text, but they are derived from it, and the relationship is closer than early deployments assumed. Membership inference attacks against embedding stores are now practical research results. For regulated data, the question of whether an embedding constitutes a copy of the original data is legally unsettled in most jurisdictions, which means the conservative posture is to assume it does.

Agent memory (the longer-term state that some frameworks maintain across sessions to give agents continuity) aggregates data from every interaction the agent has had. The aggregate is often

more sensitive than any individual interaction would be: it connects facts, identifies patterns, builds profiles. Memory systems were introduced for capability reasons and have accumulated sensitive content as a side effect.

The regulatory implications are concrete. Under GDPR, a right-to-erasure request applies to every copy of personal data the organisation holds, including the ones in vector stores and agent memory. Most deployments have no mechanism to fulfil this. Retention policies that are enforced in the source systems are typically not enforced in the retrieval and memory layers, which means data that should have been deleted after ninety days persists in the agent's knowledge long after. Cross-border data transfer restrictions that apply to primary data stores apply equally to their embeddings; most organisations cannot attest to where their vector stores live.

The corrective

Treat every agent-adjacent data store as a data store. Classify its contents, apply retention, support erasure, and audit accordingly. This is the gap that produces regulatory breaches rather than security breaches. It is the one most likely to surface not through an incident but through an audit, a subject access request, or a regulator's inquiry.

5. Gap Five: No Kill Switch, No Blast Radius Limit

The fifth gap is the one that converts the previous four from manageable risks into catastrophic ones. Unlike a compromised human insider, a compromised agent can take thousands of actions per minute. Most deployments have no rate limit, no anomaly detection, and no circuit breaker tied to agent behaviour.

The temporal asymmetry is the critical insight. A compromised human insider is bounded by human speed. Even a determined attacker with valid credentials is limited by how fast they can type, navigate interfaces, and execute actions. Detection and response playbooks were designed around these constraints: a SOC that responds within hours can contain most human-insider scenarios before the blast radius exceeds recoverable scope.

A compromised agent is not bounded by human speed. It is bounded by API rate limits, infrastructure throughput, and the transitive reach of its authorisations. An agent that can exfiltrate records at a thousand per second will have exfiltrated a million records before a conventional SOC escalation path produces a human responder. An agent induced via prompt injection to send emails will have sent tens of thousands before anyone notices the outbound queue.

The OWASP Top 10 for LLMs calls the financial variant of this risk "unbounded consumption," and the category captures both the denial-of-service and runaway-cost dimensions. The 2025 revision broadened the category explicitly because real incidents (agents stuck in tool-calling loops, agents induced to exhaust API quotas, agents producing outputs that compounded through downstream systems) had accumulated enough evidence to warrant the expansion.

Three controls required

Rate limits per agent, per tool, and per tenant. Every agent should have a budget (in API calls, in tokens, in actions per minute) appropriate to its purpose. Exceeding the budget should trigger circuit breakers, not just alerts. This is table stakes in mature deployments and absent in most first-year ones.

Behaviour-level anomaly detection. Agents have baselines. An agent that has never written to production databases suddenly writing to them is an anomaly worth surfacing. An agent whose tool-use distribution shifts dramatically from one day to the next is an anomaly worth surfacing. The telemetry required already exists in most deployments; the analytics layer does not.

A kill switch that can be exercised in seconds, not hours. Every agent deployed to production should have a documented path to being shut down: all instances, immediately, with a clear owner and a clear procedure. Most deployments we review either lack this entirely or have a kill switch that requires coordination across teams and will not complete within the window in which a compromise matters.

The Unified Picture

The five gaps do not operate independently. They compound.

Identity sprawl makes every subsequent control harder to apply, because none of them can be scoped per-agent if there is no per-agent identity. Permission creep determines how much damage a compromised agent can do, because the blast radius is the agent's authority. The tool-use trust boundary determines how compromise arrives, because indirect prompt injection through tool output is now the dominant attack vector. Memory systems determine how much data the compromise exposes, because the agent's knowledge is often broader than its nominal scope. The absence of a kill switch and rate limits determines how fast the compromise propagates, because unbounded action at machine speed is what distinguishes agentic AI from every prior insider threat.

An organisation that has closed any one gap without addressing the others has not closed the blast radius; it has moved it. Strong per-agent identity with excessive permissions produces attributable catastrophe. Tight permissions with unvalidated tool use produces injection-driven compromise within authorised scope. Good memory governance with no kill switch produces a compromise that is compliant with retention policy all the way to breach.

The integrated posture is what matters. Every agent has an identity, bounded permissions, validated tool boundaries, governed memory, and monitored behaviour with a rapid response path. No single control is novel. The integration (the discipline of applying all five simultaneously to a class of actor that is neither human nor conventional software) is where the work lives.

The Mapping: Agents as a New Actor Class

The argument of this paper, compressed: agents are a new class of actor in the enterprise, and they need the same controls that apply to the existing classes, adapted for their specific properties. The mapping is clean.

1. **Identity management,** which enterprises apply to human users, needs to be extended to agents. Every agent has an identity, an owner, a lifecycle, and a revocation path.
2. **Role-based access control,** which enterprises apply to human users, needs to be extended to agents. Every agent has a role, a minimum-viable permission set, and an escalation path for actions beyond its role.
3. **Data-loss prevention and third-party integration review,** which enterprises apply to external data flows, need to be extended to agent tool use. Every tool is a trust edge, every input crossing that edge requires validation, every pattern across multiple tools requires central reasoning.
4. **Data classification, retention, and subject-access processes,** which enterprises apply to primary data stores, need to be extended to agent memory, vector stores, and conversation histories. Every agent-adjacent data store is a data store.
5. **Security operations,** which enterprises apply to human-speed threats, need to be extended to machine-speed threats. Every agent has baselines, anomaly detection, rate limits, and a kill switch that can be exercised in the window that matters.

Five disciplines. None of them new. All of them required. The organisations that will deploy agentic AI safely over the next three years are not the ones that invent novel AI-security paradigms. They are the ones that recognise agents as a new category of actor, and extend the disciplines they already have to cover the new category. The engineering is operational, not conceptual. The work is in the doing, not the discovering.

Conclusion: The Next Eighteen Months

We expect the next eighteen months of agentic AI deployment to produce the first generation of high-profile breaches where the root cause is not a model failure but a control failure: incidents where the model did exactly what it was prompted to do, and the organisation had not constrained what it was allowed to do. We expect these incidents to follow the five-gap pattern described in this paper, with heavy representation of the identity-sprawl and excessive-permission categories in the initial wave.

We also expect that the organisations deploying agents with the integrated posture described above will experience these incidents at a materially lower rate than the industry average, and will contain the incidents that do occur within a materially smaller blast radius. Not because their agents are better. Because their controls are.

The productivity case for agentic AI is real and increasingly established. The cost case is increasingly favourable. The capability case improves monthly. None of these are in dispute, and none of them are the point of this paper. The point is that the security case is a separate workstream, and it is the one most likely to be underinvested in the race to capture the productivity gains.

The agentic blast radius is not a property of the model. It is a property of the controls (or the absence of them) that the deploying organisation puts around the model. Those controls are not mysterious. They are the controls the enterprise already applies to human actors, adapted for a class of actor that operates at machine speed without intent. Extending them is the work.

The alternative is to deploy autonomous systems with shared identities, excessive permissions, unvalidated tool use, ungoverned memory, and no kill switch, and to discover the consequences at the speed the agents themselves operate.

References

1. Gartner (2025). “Emerging Technology Impact Radar: Agentic AI.” Gartner Inc.
2. IBM Security & Ponemon Institute (2025). “Cost of a Data Breach Report 2025.”
3. OWASP Foundation (2025). “OWASP Top 10 for Large Language Model Applications, v2025.”
4. NIST (2024). “AI Risk Management Framework: Generative AI Profile.”
5. European Parliament (2024). “Regulation (EU) 2024/1689: Artificial Intelligence Act.”
6. Anthropic, OpenAI, Google DeepMind (2024–2026). Published safety research on agentic systems and indirect prompt injection.
7. MITRE ATLAS (2025). Adversarial Threat Landscape for AI Systems: agentic extensions.
8. Cloud Security Alliance (2025). “Non-Human Identity Management: State of the Industry.”

This whitepaper is part of 01′s research series on enterprise AI posture. 01 helps enterprises design agentic systems that are safe to deploy at scale, from identity and authorisation architecture to tool-use governance and operational readiness.

www.01ltd.com/contact