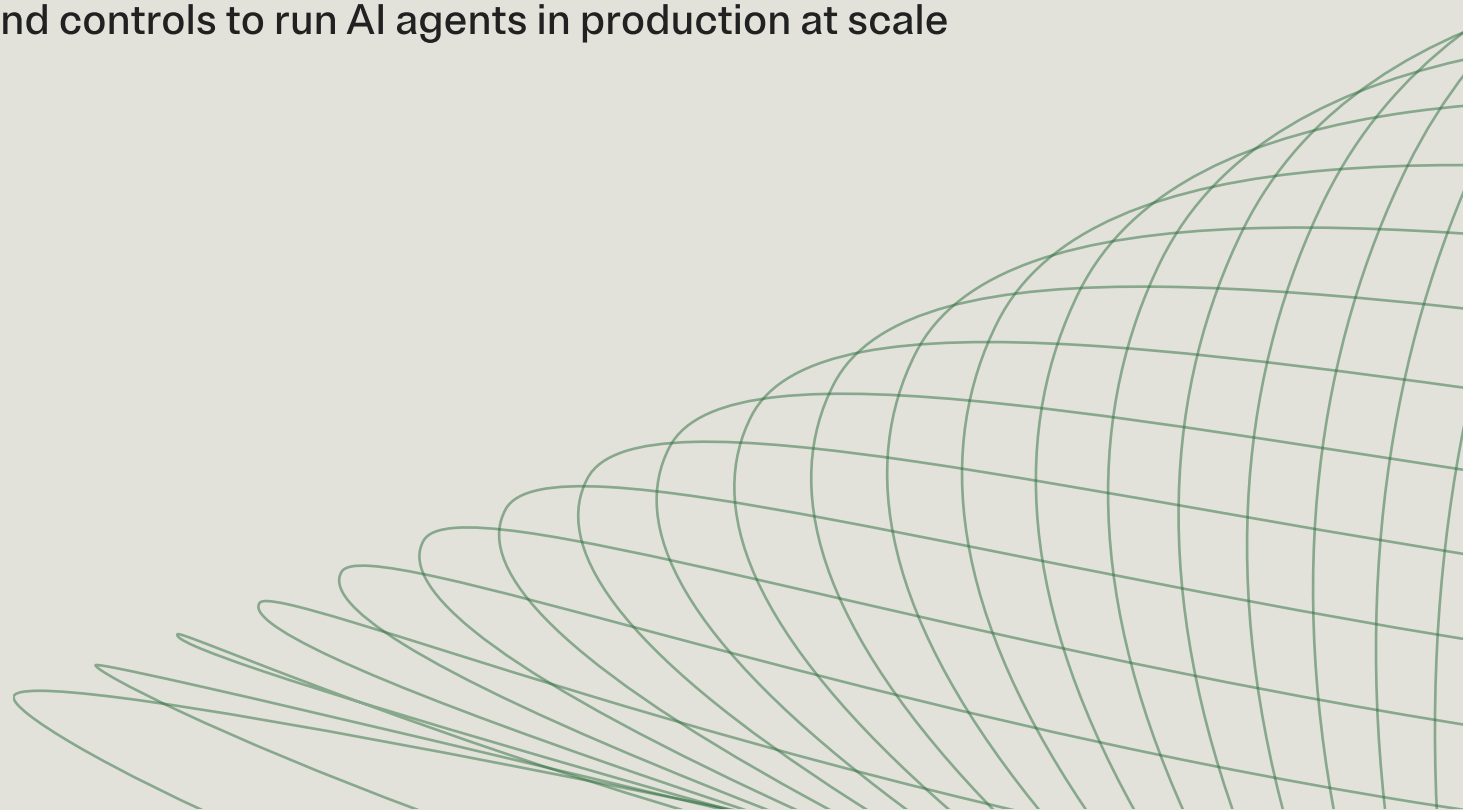
STACKLOK

AI Control Plane Buyer's Guide for Financial Services

Key considerations for leaders who need the governance and controls to run AI agents in production at scale



Contents

Over the past year, we have worked with financial services firms to drive deeper adoption of AI agents and MCP servers. This guide distills what they told us matters most, and what separates governance theater from genuine control.

1. Why Financial Services Leaders Must Act Now

AI agents have already changed how financial services firms operate. They are in production and busy querying risk databases, drafting credit memos, pulling data from trading platforms, filing compliance tickets, and more. Financial services firms are connecting agents to their enterprise systems so they can take action. That means that they are touching customer data, credit information, and regulated records.

This transformation is happening with or without a governance framework in place. For example, teams are standing up MCP servers before any policy exists to govern them, because there is no rule yet that says they cannot. Platform engineers are working quickly to implement controls and patterns, and security architects are bringing deployments into compliance.

This creates a specific and urgent problem for senior leaders in financial services. You are not just managing technology adoption, but regulatory exposure. A misconfigured MCP server that gives an AI agent unaudited write access to a risk database, or that logs tool calls against a shared service account rather than the individual who triggered the task, is a potential regulatory event. The FCA, SEC, PRA, and their international counterparts are paying close attention to how firms govern AI systems that touch regulated processes.

What makes this moment particularly consequential is the compounding effect of delay. Every week without a governance framework is a week in which more ungoverned deployments accumulate, more shadow integrations embed themselves into developer workflows, and more technical patterns calcify in ways that will be painful and expensive to retrofit.

The leaders who resonate most with this challenge are the ones who own the board-level pressure to demonstrate an AI strategy. They believe in the technology and understand that to move with speed, they must have the right underlying control plane.

Firms that establish a genuine AI control plane now will define how their organization runs agentic AI for the next decade. Those who wait will spend that time in remediation: unpicking shadow deployments, failing audits, and trying to retrofit controls onto an estate that was never designed to be governed.

2. The Reality on the Ground: What We See at Financial Services Firms Today

The picture we encounter at financial services firms evaluating AI infrastructure is remarkably consistent:

MCP sprawl has already started

Teams are standing up MCP servers before any governance rules exist. At one insurance conglomerate with 146 subsidiaries, MCP deployments happened because there was no policy in place. At a major cybersecurity firm with around 8,000 employees, distributed teams are independently building agents and MCP servers with almost no cross-team visibility.

Authentication is improvised and audit trails are broken

Most early MCP implementations rely on API keys baked into configuration files, shared service principals, or personal access tokens with no rotation policy and no meaningful audit trail. When a downstream system like a data warehouse logs a tool call, it records a service account, not the analyst, engineer, or agent that actually triggered it. That is not a defensible audit position for any regulated firm.

Security teams are blocking rather than enabling

The right response to security team concerns is to give them the audit logs, access controls, identity passthrough, and policy framework they need to say yes. Without that, every new MCP server requires a bespoke review, adoption slows to a crawl, and developers route around controls.

Governance is being invented ad hoc

Patterns are being defined on the fly, without prior art to draw from. The MCP specification itself is still maturing. Patterns established today need to hold as server counts scale from five to five hundred, and for financial services firms those patterns need to survive regulatory scrutiny from day one.

3. The Distinct Stakes for Financial Services

Financial services firms face a version of the AI governance problem that is materially harder than most other sectors. Several factors compound the challenge:

Regulatory exposure is direct and personal. Audit failures, data boundary violations, and inadequate access controls are regulatory events. Regulators including the FCA, SEC, PRA, and others are increasingly focused on AI governance. The senior leaders accountable for those failures can be personally exposed.

Organizational complexity works against you. Most large financial services firms operate across multiple legal entities, jurisdictions, and regulatory environments simultaneously. A governance model that works for a single business unit may not work across that landscape without deliberate multi-tenancy and jurisdictional controls.

Data sensitivity is non-negotiable. AI agents connecting to internal systems in financial services are connecting to some of the most sensitive data that exists: PII, trading data, credit information, actuarial models, customer communications. A misconfigured MCP server is a potential data breach.

The threat surface is evolving faster than controls. The combination of agentic AI and MCP creates new attack surfaces: prompt injection through tool calls, credential exfiltration via malicious MCP servers, uncontrolled access to metered or rate-limited downstream systems.

Procurement and approval cycles lag the technology. By the time formal procurement processes complete, developers have already built workarounds. The firms handling this best are using open source evaluation paths to build internal confidence before a procurement conversation.

4. The Core Problems an AI Control Plane Must Solve

Before evaluating any platform, it is worth being explicit about the problem surface. Five categories of challenge consistently emerge as financial services firms try to scale AI agents from experimentation to production:

Identity and audit passthrough is the hardest technical problem. When an AI agent calls a tool through an MCP server, which identity does the downstream system see? In most current implementations, it sees a shared service account. To avoid breaking your audit trail, it needs to identify the analyst, engineer, or automated pipeline that triggered the call.

Governance and policy enforcement at scale is a fundamentally different problem from governance at five servers. Governing fifty MCP servers spanning multiple teams, business units, legal entities, and deployment environments requires a policy model that can express fine-grained controls. Those policies need to be declarative, version-controlled, auditable, and enforceable at runtime without redeployment.

Discoverability without shadow IT is a genuine tension. The harder you make it to consume MCP servers through official channels, the more developers will route around those channels. Without a curated registry and an approval workflow, you have no visibility into what is running, and no way to enforce supply chain security or data boundary controls. The right answer is a registry that is frictionless enough to be the default path.

Operational maturity and observability are non-negotiable in regulated environments. You need to know what MCP servers are running, who is calling them, what tools are being invoked, and whether those invocations are succeeding or failing. That telemetry needs to flow into your existing SIEM and observability stack, and audit logs need to be retained, queryable, and exportable in formats that will satisfy regulatory inspection.

Deployment flexibility across heterogeneous and air-gapped environments is a hard requirement, not a nice-to-have. Financial services infrastructure is rarely homogeneous. You likely have Kubernetes clusters across multiple clouds, on-premises infrastructure, air-gapped or isolated environments for sensitive workloads, and a mix of local developer tooling. A platform that only operates in one deployment model will create governance gaps in the others.

5. The Capability Checklist

Use this framework when evaluating vendors. These requirements reflect the real challenges that financial services firms are working through as they scale AI agents to production. Where capabilities are table stakes, they are noted as such. Where they represent a higher bar that separates serious enterprise platforms from lighter-weight alternatives, that distinction is called out.

Section 1: Identity, Authentication & Audit

This is the category where the gap between credible enterprise platforms and lightweight alternatives is widest, and where the regulatory stakes for financial services are highest. Treat any vendor that cannot give detailed, concrete answers here with significant skepticism.

Capability	Why it Matters	What to Require
Enterprise IdP integration (Okta, Entra ID, Ping, Google)	Developers must authenticate through your existing identity provider. Separate credentials for MCP access create security gaps and adoption friction.	Require native OIDC/OAuth SSO with your IdP, with no stored API keys or personal access tokens in client configurations. If you are on Entra ID, verify the vendor understands Entra's dynamic client registration limitations.
Per-user identity passthrough to downstream systems	When an AI agent calls a tool, your data warehouse, ticketing system, or risk platform, you should log the actual user's identity, not a shared service principal.	Require token exchange flows (OAuth on-behalf-of / RFC 8693) that preserve and propagate end-user identity to downstream systems. Ask for an end-to-end architecture walkthrough of exactly how this works.

Short-lived, scoped tokens with automatic rotation	Static credentials are a security liability. Each tool call should operate under a scoped, short-lived token appropriate to the action being performed.	Require that the platform issues short-lived tokens per session or per request, with no requirement for developers to manage credential rotation manually.
Non-human identity support for agents	Automated agentic workflows operate without a human in the loop. Your governance model needs a story for workload identity. Unmanaged service accounts for agents are a significant vulnerability.	Require support for workload identity patterns including SPIFFE/SPIRE-based JWT authentication for non-human agents running in automated pipelines and CI/CD workflows.
Read-only and granular operation-level scoping	Legal, compliance, and security teams will frequently need to allow tool access while restricting destructive operations.	Require the ability to scope tool permissions by operation type (read, write, delete) at the policy level, not just at the server access level. Verify enforcement happens at the gateway, not just in client configuration.
SIEM-ready audit log export	Security operations teams need MCP audit data in the same place they monitor everything else. Isolated audit logs need to feed your SIEM to be ready for regulatory inspection.	Require log export in formats compatible with your SIEM platform (Splunk, Elastic, Microsoft Sentinel, etc.). Confirm logs are tamper-evident, include full identity and tool call context, and can be retained for the periods your compliance framework requires.

Section 2: Governance, Policy & Compliance

Capability	Why it Matters	What to Require
Curated server registry with approval workflows	Without a registry, you have no visibility into what MCP servers are running across the organization. With a registry that is painful to use, developers route around it and your shadow IT problem grows.	Require a registry that supports metadata (owner, security review date, data classification, approved client list) and a vetting workflow that integrates with your existing approval processes. The registry must be easy enough to use that it is the default.
Supply chain security for MCP servers	MCP servers sourced from the public ecosystem (npm packages, GitHub repos) represent a real supply chain risk.	Require CVE scanning and SLSA attestation for all servers in the registry. Ask whether the vendor maintains hardened versions of high-usage community MCP servers (GitHub, Atlassian, Databricks, Salesforce, etc.) with ongoing security patch commitments.

Granular, declarative policy engine	Access control at the server level is insufficient for production. In financial services, you need tool-level, claim-based, and role-based policies that are readable, version-controlled, and testable.	Require a policy engine that supports RBAC, ABAC, and claim-based authorization at both the server and tool invocation level. Policies should be declarative code. Verify compatibility with your existing policy framework (Cedar, OPA/Rego, or equivalent).
Policy changes without redeployment	Governance needs to respond to incidents in real time. If changing a policy requires a full redeployment cycle, your incident response capability is compromised and your ability to contain a live security event is constrained.	Require that policy changes take effect at runtime without server or gateway redeployment. This is a meaningful differentiator that many platforms cannot deliver.
Multi-tenant, jurisdictional controls	For FS firms operating across multiple legal entities, business units, or geographies, governance that works at the organization level but cannot be scoped to subsidiary or jurisdiction level is inadequate.	Require namespace-level isolation with RBAC controls that enable different business units or subsidiaries to have distinct server catalogs and access policies. Ask specifically about multi-tenancy and parental oversight models.
Client-side enforcement hooks	Governance that only exists at the server layer can be bypassed by a developer who configures their own MCP server locally. Enforcement at the client layer closes that gap.	Require hooks-based enforcement for your approved IDE clients (VS Code, Cursor, Windsurf, Claude Code). Ask specifically which clients are supported today, and get explicit answers on any gaps.
Tool filtering and virtual MCP server composition	As your MCP footprint grows, individual servers expose too many tools for effective or safe agent use. A server exposing 1,200 AWS tools to every developer is both a governance risk and a context window problem that degrades agent performance.	Require the ability to define virtual MCP servers tuned per team, role, or use case that expose only the tools appropriate to that context. This is a meaningful differentiator that most lightweight platforms do not support.

Section 3: Runtime Security & Deployment

Capability	Why it Matters	What to Require
Kubernetes-native operator	MCP servers need to be managed like any other production workload with CRDs, Helm charts, operator-based lifecycle management, and native scheduling.	Require a native Kubernetes operator with CRD-based server lifecycle management. Verify support for your specific distribution (EKS, AKS, GKE, OpenShift, or on-premises).
Container isolation per server	Each MCP server should run in its own isolated container with minimal default permissions. Shared runtimes create blast radius risk — a compromised or misconfigured server in a shared runtime can affect all others.	Require per-server container isolation with configurable network egress and filesystem access controls. For high-sensitivity environments, ask specifically about support for hardware-isolated microVM runtimes (Kata Containers, Firecracker).

Air-gapped and fully self-hosted deployment	Many regulated financial services environments cannot route data through external SaaS platforms. This is a hard requirement, not a preference. Any required SaaS component or external callback is a deal-breaker.	Confirm the platform is fully self-hostable with no required SaaS components or external callbacks. Verify this is available today — not on the roadmap.
High availability and failover	Production AI control plane infrastructure cannot be a single point of failure. Agentic workflows that depend on MCP availability will fail in ways that are visible to end users and business stakeholders if the infrastructure is not resilient.	Require multi-replica deployment support and documented failover behavior. Ask about circuit breakers at the gateway layer.
Local development support with policy enforcement	Developers will run MCP servers locally during development and prototyping. That is legitimate and should be supported — but local execution must still enforce registry policies and emit telemetry. Local deployment should not mean ungoverned deployment.	Require that local execution modes remain under central policy control and still report usage telemetry to your observability stack. The deployment model should not determine the governance model.

Section 4: Observability & ROI Measurement

You cannot govern what you cannot see, and you cannot justify AI investment to a board or regulator without evidence of what is happening.

Capability	Why it Matters	What to Require
OpenTelemetry-native telemetry	You already have an observability stack. You should not have to run a parallel one for AI agents. MCP telemetry must integrate with existing infrastructure.	Require OLTP-compatible traces, metrics, and logs following official OpenTelemetry MCP semantic conventions. Verify export to your existing backend (Datadog, Splunk, Grafana, Dynatrace, Prometheus, etc).
Baseline and anomaly visibility	A SIEM integration is most useful when you have a baseline of normal behavior. The platform should produce telemetry granular enough to surface deviations like compromised credentials, unexpected agent behavior, or out-of-policy tool calls.	Require sufficient telemetry granularity to establish per-user, per-team, and per-server usage baselines over time. This is the foundation for detecting compromise or policy violation at runtime.
Usage attribution for cost chargeback	As AI agent usage scales across business units, finance and engineering leadership will need to attribute usage and cost to specific teams, projects, or regulatory entities (particularly relevant in FS firms with distinct P&L ownership).	Require telemetry granular enough to support chargeback models: tool call counts, token consumption estimates, and user-level attribution available in the telemetry stream.

Section 5: Developer Experience & Adoption

The best AI control plane platforms make the governed path the easy path.

Capability	Why it Matters	What to Require
Self-service discovery portal	Developers need to find approved MCP servers without filing a ticket or waiting for an administrator. A curated, searchable catalog is the difference between a registry that gets adopted and one that gets ignored.	Require a self-service portal developers can access without admin assistance, supporting both a human-readable UI and a machine-readable API for programmatic integration.
Automatic IDE client configuration	Requiring developers to manually configure each IDE client is friction that slows adoption and creates configuration inconsistency across the estate.	Require automatic client configuration: when a developer gains access to a server, their IDE should be configured automatically, not through a manual YAML edit.
Context optimization	As MCP server counts grow, context window bloat degrades agent performance and burns tokens unnecessarily. Exposing 50 tools to a model that needs 3 produces worse results and is harder to govern.	Require a gateway-level context optimizer that dynamically surfaces relevant tools based on the task, rather than flooding the model with every available tool on every request.
Composite tool and workflow support	Real enterprise workflows require sequencing multiple tool calls in a defined order with transaction semantics. Leaving this to the LLM to figure out produces inconsistent, difficult-to-audit results.	Require the ability to define declarative composite tools that chain multiple underlying tool calls into a single, deterministic workflow. Ask for a demo with a realistic multi-system workflow.
Support for both local and remote server models	Some servers should run locally on the developer's machine. Others should run centrally in Kubernetes. Both deployment models are valid, and the platform must govern both the same way.	Require that local and remote server deployments are subject to the same registry policies, the same telemetry requirements, and the same identity controls.

Section 6: Commercial & Vendor Considerations

Capability	Why it Matters	What to Require
Open source core with commercial enterprise layer	Open source lets you evaluate, prototype, and build internal confidence without a procurement conversation, which can be valuable in financial services environments where formal procurement is slow.	Require an Apache 2.0 (or equivalent permissive) licensed core that includes the runtime, registry, and gateway. Enterprise tiers should add hardening, support SLAs, and convenience features.

Production support SLAs	When AI control plane infrastructure becomes business-critical, you need a vendor with defined escalation paths and response time commitments.	Require defined SLA tiers with escalation paths. Confirm support coverage matches your operational hours and that critical incident response times are contractually defined.
Outcomes-oriented engagement model	Delivering software and walking away is not adequate for infrastructure this new. The most successful deployments involve vendor engineering working alongside your team to implement, validate, and iterate.	Ask specifically about forward-deployed engineering availability. Is there a dedicated deployment engineer for the initial rollout? Will engineers write code alongside your team? A vendor who can embed is worth significantly more than one who simply hands over the keys.
Modular architecture ('use what you need, bring what you have')	Your organization has already invested in identity systems, policy engines, registries, and observability platforms. A platform that requires you to replace all of that is a much harder internal sell.	Require explicit confirmation of which components are swappable. Can you use your existing OPA/Rego policies instead of Cedar? Can the registry coexist with your existing artifact management? Every confirmed modularity point de-risks the procurement conversation.

6. Prioritizing Your Evaluation

Not every capability carries equal weight for every organization. Use the following framework to sequence your evaluation based on your current situation.

If your primary concern is regulatory and audit readiness

Lead with per-user identity passthrough, SIEM-ready audit log export, configurable retention, and granular policy enforcement. These are the capabilities that allow you to walk into a regulatory examination or internal audit with credible answers. Nothing else matters if you cannot demonstrate a clean, attributable audit trail.

If your primary concern is security team buy-in

Lead with authentication and identity passthrough, client-side enforcement hooks, supply chain security, and container isolation. These are the capabilities that allow security teams to say yes to broader adoption rather than defaulting to binary block decisions. A platform team's job is often to get security comfortable enough to unlock rollout.

If your primary concern is developer adoption without shadow IT

Lead with the self-service discovery portal, automatic client configuration, and the open source evaluation path. Developers who can get value quickly become internal advocates. Developers who hit friction create the shadow deployments you are trying to prevent. The governed path must also be the easy path.

If your primary concern is scaling from your current footprint

Lead with the Kubernetes operator, multi-tenant support, virtual MCP server composition, and context optimization. The patterns you establish for five servers need to hold at fifty — and in a regulated environment, they need to hold up under audit from day one. Platforms that seem adequate at small often cannot make that transition gracefully.

If your primary concern is justifying AI investment to the board

Lead with observability, usage telemetry, and tool call performance instrumentation. You cannot prove return on investment for something you cannot measure. Per-developer and per-team usage data, adoption trend visibility, and tool call success rates are the inputs to a credible ROI narrative.

7. What the Market Actually Looks Like

The AI control plane and MCP platform market is early. Several vendors have entered the space, and it is worth being clear-eyed about where they focus and where they fall short when evaluated against financial services requirements.

Lightweight gateway tools can proxy MCP traffic and provide basic routing, but they typically lack the runtime management, supply chain security, multi-tenant governance, and identity depth that regulated production deployments require. They are useful for prototyping. They are not adequate for governing a multi-team MCP estate under regulatory scrutiny.

Developer-tooling platforms that have added MCP support often treat it as a feature rather than a first-class primitive. Their governance models are designed for developer convenience, not enterprise security posture. Their audit trails are thin, their identity integrations are limited, and they were not built with regulatory inspection in mind.

Cloud provider offerings address deployment but typically lock you into a single-cloud architecture, lack vendor-neutral flexibility, cannot govern a hybrid local-plus-remote server landscape, and do not provide the identity passthrough and policy depth that regulated environments require.

The right platform will be purpose-built to solve the enterprise AI governance problem: a Kubernetes-native operator, a full identity and policy stack, supply chain security, multi-tenant controls, and an open source core that lets you evaluate without asking for permission. It treats governance not as a constraint on developer productivity, but as the foundation that makes broader, safer adoption possible.

The questions in this guide will help you tell the difference.