Model Context Protocol

The Missing Layer in Securing Non-Human Identities

by Lalit Choda (Mr NHI)

The cybersecurity perimeter isn't just about human users or login screens anymore.

Instead, it's moving toward something a lot more complex and maybe even more risky: Non-Human Identities (NHIs) that act on their own, make choices, and have control over various systems.

AI models like Claude or ChatGPT now perform far more than they were originally trained for. Today, NHIs outnumber human ones by a wide margin, with LLM agents and software supply chain bots leading the pack — it's a ratio of 25 to 50 times! But as these digital entities keep growing, there's a big gap in how we manage them. We've got the hang of authenticating users. We still haven't figured out how to manage machines that can think and act on their own.

So, this is where the Model Context Protocol (MCP) steps in.

MCP isn't just a buzzword; it's an up-and-coming protocol designed to provide digital entities with a structured behavioral context. It suggests moving away from identity-based access to a system that enforces execution based on context, tying what a machine can do to the where, when, and why of its actions.

What Exactly Is Model Context Protocol?

The Model Context Protocol, or MCP, is a structured and open protocol that aims to link large language models (LLMs) with tools, data, and services in a standardized and secure manner.

So, when an AI model like Anthropic's Claude or OpenAI's GPT needs to do things beyond what it knows—like checking a database, calling a REST API, or getting private data—it can use MCP to ask for access and get a response from a trusted server. But MCP is more than just connections. It gives you the lowdown on what's happening: what the model is up to, what tools it can use, who the user is, what data is being accessed, and the policy guiding the action.

To put it simply, MCP serves as the reliable link and translator between an AI agent and everything beyond its reach. It makes sure that models work within clear boundaries, with the right context, accountability, and policy enforcement. Plus, it guarantees that every decision or action taken by an NHI includes:

1. The intended behavior and model state
2. The policy scope (what's allowed and what's not)
3. The source of invocation (who or what triggered the action)
4. And the environmental metadata (time, workload type, data boundaries)

MCP vs Traditional IAM: What's New?

Feature	Traditional IAM	Model Context Protocol (MCP)
Who gets access?	Regular users or service accounts	Smart AI agents and models
How is access given?	Based on fixed roles and predefined rules	Based on what the model is doing and the context it's in
Who decides the rules?	A system that uses roles and permissions (RBAC/PBAC)	A system that understands intent and adjusts based on context
Who starts the action?	A system that uses roles and permissions (RBAC/PBAC)	The AI can act on its own, but only after verifying the context
What gets recorded?	Just the user's actions	Everything — what was done, why it was done, and which tool was used
How detailed is access?	Broad permissions like "read-only" or "admin"	Very specific — like "allow only this model to access just this one dataset for this task"

MCP = Identity + Execution Context + Behavioral Constraints.

MCP takes things a step further than traditional IAM systems. While those systems focus on identifying who an entity is, MCP asks, "Should this action be allowed right now, in this context, and with this level of trust?"

How MCP and NHIs Intersect

AI models that interact with systems, like retrieving sensitive records are effectively acting as NHIs. That means they must be:

- Identified: Who or what is the agent?
- Scoped: What can it do?
- Monitored: What has it done?

MCP provides the structure for these controls. It allows organizations to delegate actions to AI agents safely, while enforcing security boundaries and business logic around what those agents can see or do.

Through MCP:

1. NHIs powered by LLMs can access tools only when explicitly allowed

2. Context (user session, role, task) is embedded with every action

3. Organizations retain full control over tool servers, data policies, and logging

The NHI Problem

Back in the day, identity was just about having a username and password. For NHIs, identity feels a bit abstract. These Non-Human Identities (NHIs) have become the main players in many organizations, actually outnumbering human users by a significant margin. You've got service accounts, API keys, LLM models, and AI agents in the mix.

What's the issue? So, these NHIs are:

- Invisible, since they're not really monitored like human users
- Powerful because they have broad permissions
- Poorly governed, often having stale credentials or no clear owner.

MCP shifts the discussion from "what identity is this?" to "what context is this action happening in?" That shift really changes the game.

MCP's Approach to Tackling NHI Issues

The Model Context Protocol (MCP) provides a fresh approach: it focuses on securing NHIs by incorporating context, control, and traceability into each action they take. Let me break it down for you:

Contextual Execution - MCP makes sure that an NHI can only work within its intended model scope. So, what this means is that an AI agent that's been trained for documentation just can't jump in and start interacting with financial systems. The context of execution just doesn't permit that.

Policy Binding - Rather than just linking access rules to an identity or endpoint, MCP applies behavioral policies at the model context level. This lets NHIs be guided not just by their identity, but also by their actions and the reasons behind them.

Auditability - Every action taken by NHI through MCP is logged with complete context: intent, origin, scope, and response. So, what this means is that the choices made by autonomous systems can be looked back on, explained, and examined. This is really important for building trust and ensuring compliance.

Challenges

Every transformation comes with its own set of challenges. To adopt MCP, we need to tackle:

- Context Modelling - Defining accurate boundaries for complex systems can be quite a challenge, especially when it comes to multi-agent or hybrid cloud environments.

- Legacy Compatibility - A lot of the IAM systems out there weren't really built to handle contextual enforcement. Getting MCP to work in these environments requires some integration effort.

- Standardization - For MCP to really mature, it's going to need to work well across different platforms. If we don't have common tool servers or policy schemas, there's a real risk that fragmentation could undermine its potential.

For a secure future with NHIs, we can't just depend on old-school human access controls. As machines get smarter and start making decisions, it's important that the way we govern them adapts too. The Model

Context Protocol provides a way to move ahead. It's not a quick fix, but it definitely marks a key change from fixed identities and wide-ranging permissions to more flexible, context-based policy enforcement. If it's designed well, MCP could turn into the digital system that makes NHIs predictable, safe, and accountable.

The future of cybersecurity is moving away from just usernames and passwords. It's going to be influenced by the model's identity, the scope of the task, and the limits on behavior. MCP is set to be a key building block for Zero Trust in machine-driven infrastructure. When it comes to AI assistants handling workflows or robotic process automation in finance, it's all about earning trust through actions rather than just relying on credentials.