LOUST-PRO/outlook-mcp-suite

Source: LOUST · **Status:** Shipped · **Period:** 2025-03 (active)

Twenty-two-tool MCP bridge over Microsoft Graph for Outlook mail, calendar, and contacts — read-only by default, explicit opt-in to mutate.

The challenge

Agentic loops need calendar and mail context to be useful, but giving an LLM free reign over a real inbox is a recipe for a bad day. I needed a bridge that exposed Microsoft Graph capabilities to agents without exposing the inbox itself.

My role

I built a 22-tool MCP bridge over Microsoft Graph where every tool is read-only by default, mutating tools require explicit opt-in, and the broker surfaces the actual API call in response metadata.

What I did (3–5 decisions)

1. Read-only by default

Why: An LLM with read-only access is useful; with read-write access it is a liability. Read-only is the safe default that still delivers the context an agent needs.

Tradeoff: Workflows that require mutation go through explicit opt-in. The opt-in is the chokepoint that keeps unexpected writes from happening silently.

2. Intent-filtered broker pattern

Why: The broker surfaces what an agent actually asked for in the response metadata. The agent (and the reviewer) can confirm what was fetched without re-running the request.

Tradeoff: Slightly higher per-call overhead. The audit clarity earns the extra latency.

3. Narrow, annotated tools

Why: Each tool does one thing and announces what it touches. Twenty-two narrow tools beat one mega-tool because the failure surface of any single tool is bounded.

Tradeoff: Twenty-two tools to maintain instead of one. Per-tool audit clarity wins because debugging a mega-tool is where most of the time goes.

What changed

Inbox safety

Baseline: Exposed — agent had free reign

Isolated — broker mediates every call

Agent context coverage

Baseline: About 30% (manual calendar and mail lookups)

About 95% (MCP-mediated access to the full graph)

Evidence: estimate, not measured

Audit chain

Baseline: Opaque — no record of which calls fired

Per-call intent log, replayable

Overall tradeoffs

I gave up some agent flexibility for inbox safety. The safety is the whole point — the day an LLM sends mail on my behalf without my knowledge is the day I turn the bridge off.

Lessons

MCP bridges earn their keep by being the bottleneck that prevents bad agentic behavior. A bridge with no chokepoint is just a fancier HTTP client.

Stack

Go | MCP | Microsoft Graph

Repository

github.com/LOUST-PRO/outlook-mcp-suite

Evidence

- [repo] github.com/LOUST-PRO/outlook-mcp-suite