MCP Readiness Checklist

This is a go or no-go gate you run before you connect an MCP server to an assistant, so a useful integration does not quietly become a path for data to leak or for a stranger to act with your access. It is about the connection, the trust you are extending, and the blast radius if a token leaks, separate from whether a single task is safe to automate (that is the Agent Readiness Checklist). Run it the first time you enable a server, and again whenever the server, its scopes, or its publisher change.

Value and fit

- [ ] You can name the specific task this server enables, and why it beats pasting the data by hand.
- [ ] No narrower option (a read-only scope, a single system, an existing tool) would do the same job with less access.
- [ ] The server is worth the standing risk of being connected, not just useful in the moment.

Access and authorization

- [ ] The server requests the narrowest scopes for the job, read-only where possible, one system rather than all of them.
- [ ] The server accepts only tokens issued for itself and rejects tokens whose audience is a different service.
- [ ] The server does not pass your token through to a downstream API; it uses its own scoped credential for upstream calls.
- [ ] Remote servers use OAuth 2.1 with PKCE over HTTPS, with exact redirect URI matching (no wildcards).
- [ ] A per-client consent screen names the client, the third-party scopes, and where tokens will be sent, and you actually saw it.
- [ ] Each server has its own scoped credential; no token is shared across servers.

Trust and supply chain

- [ ] You verified the publisher and the exact package name, and it is not a typosquat of a popular server.
- [ ] You read or scanned the server code and its dependencies, and pinned an exact version.
- [ ] You will re-review and re-pin when the version changes, so a silent tool-definition swap (a rug pull) cannot slip through.
- [ ] Tool descriptions and parameter schemas were inspected for hidden instructions before approval.

Containment and runtime safety

- [ ] A local server runs over stdio or a restricted socket, in a sandbox with only the files and network it needs.
- [ ] You saw the full startup command, untruncated, before it ran, and it does not touch SSH keys, system files, or unexpected network endpoints.
- [ ] Tool descriptions and tool results are treated as untrusted, attacker-controllable text that can carry prompt injection.
- [ ] No high-risk action runs automatically on the strength of tool output; a human reviews it first.
- [ ] The server cannot reach internal IP ranges or the cloud metadata endpoint at 169.254.169.254.

Blast radius and ownership

- [ ] You know what one leaked or stolen token would expose, and the answer is bounded, not "everything".
- [ ] You can revoke or rotate the credential quickly without breaking unrelated workflows.
- [ ] One person owns this connection, and updates are re-reviewed before they land in a shared config.
- [ ] Tool-call logging is on from the first day, so an incident can actually be investigated.

Decision

- [ ] Every safety line is a clear yes; any no or unknown blocks the connection until it is resolved.
- [ ] The decision is recorded with its scopes, conditions, and a trigger for when to re-check.