COMMUNITY SECURITY RESOURCE

MCP Security Checklist

A practical, community-maintained security baseline for teams building and deploying Model Context Protocol servers and AI agent infrastructure. Vendor-neutral. Free to use. Open to contributions.

6	36	2	3
SECURITY DOMAINS	TOTAL CONTROLS	CRITICAL CATEGORIES	AUDIENCE TIERS

QUICK START

Top 10 Controls

01 Never expose MCP over the public internet without mTLS or equivalent.

02 Scope every tool to the minimum necessary permissions.

03 Validate and sanitize all inputs before they reach tool execution.

04 Log every tool invocation with the originating session context.

05 Set rate limits on both the MCP server and any downstream APIs it calls.

06 Treat agent sessions as untrusted by default — validate intent, not just tokens.

07 Separate read and write tools; require explicit approval for write ops in sensitive contexts.

08 Rotate credentials used by MCP servers on a defined schedule.

09 Monitor for behavioral anomalies: unusual tool chains, high-frequency calls, off-hours access.

10 Conduct a tool inventory review before every production deployment.

FULL CHECKLIST

Security Controls

Authentication & Authorization

CRITICAL · 10 controls

- [ ] Require authentication on all MCP endpoints
  Unauthenticated tool invocation allows any caller to execute tools.
  auth-001
- [ ] Use short-lived tokens for agent sessions
  Long-lived static keys increase exposure window on compromise.
  auth-002
- [ ] Implement mTLS for server-to-server MCP communication
  Mutual TLS ensures both parties are authenticated in transit.
  auth-003
- [ ] Do not embed credentials in agent prompts or tool descriptions
  Agents can leak credentials in responses or logs.
  auth-004
- [ ] Use a secrets manager for MCP server credentials
  Environment variables are frequently exposed in logs or config dumps.
  auth-005
- [ ] Apply least-privilege scoping to every tool
  Overprivileged tools expand blast radius on compromise.
  auth-006
- [ ] Separate read-only and write/execute tools
  Require explicit elevated authorization for destructive operations.
  auth-007
- [ ] Enforce per-agent identity on tool invocations
  Shared credentials prevent individual agent traceability.
  auth-008
- [ ] Validate tool-level authorization, not just server-level auth
  Gateway-only checks can be bypassed via direct MCP calls.
  auth-009
- [ ] Restrict which tools are visible to which agents
  Tool discovery itself is an attack surface.
  auth-010

Input Validation & Prompt Injection

CRITICAL · 6 controls

- [ ] Validate structure and type of all tool inputs before execution
  Unvalidated inputs can cause unexpected behavior or injection.
  input-001
- [ ] Define and enforce input schemas for every tool
  Schema enforcement rejects malformed and adversarial inputs early.
  input-002
- [ ] Set maximum length limits on all string inputs
  Unbounded inputs can cause resource exhaustion.
  input-003
- [ ] Separate data from instructions at the tool layer
  Mixing data and instructions enables prompt injection via retrieved content.
  input-004
- [ ] Validate output of tool results before passing back to model
  Tool responses can contain adversarial instructions targeting further agent calls.
  input-005
- [ ] Require explicit confirmation for irreversible actions
  Reduces impact of successful prompt injection or misconfiguration.
  input-006

Tool & Resource Exposure

HIGH 5 controls

- [ ] Maintain a formal inventory of all exposed tools
  You cannot secure what you haven't catalogued.
  tool-001
- [ ] Review tool inventory before every deployment
  Development tools frequently carry over-permissive access into production.
  tool-002
- [ ] Expose only tools an agent needs for its current task
  Minimal tool surface limits blast radius of compromised sessions.
  tool-003
- [ ] Use separate MCP instances for different agent roles
  Mixing admin and read-only tools enables privilege escalation.
  tool-004
- [ ] Apply egress filtering on all external API calls from MCP tools
  Unfiltered egress enables data exfiltration via tool-initiated calls.
  tool-005

API Session Security

HIGH · 5 controls

- [ ] Assign a unique session ID to every agent interaction
  Session IDs are required to trace multi-step tool chains.
  session-001
- [ ] Enforce session timeouts
  Indefinite sessions increase the window for token abuse.
  session-002
- [ ] Apply rate limits per session and per tool
  Server-level rate limiting alone is insufficient for agentic workloads.
  session-003
- [ ] Alert on sustained high-frequency tool calls from a single session
  Indicates runaway loops, abuse, or compromised agents.
  session-004
- [ ] Log full request context for every tool call
  Incomplete logs prevent incident reconstruction.
  session-005

Monitoring & Observability

HIGH 5 controls

- [ ] Log every tool invocation with session ID, agent identity, tool name, and timestamp
  Minimum required data for incident response.
  mon-001
- [ ] Log tool chain sequences within sessions
  Unusual chains are a primary detection signal for agentic threats.
  mon-002
- [ ] Centralize MCP logs into SIEM or log aggregation platform
  Siloed logs are inaccessible during a server compromise.
  mon-003
- [ ] Retain logs for a minimum of 90 days
  Agentic attack chains can take time to fully manifest.
  mon-004
- [ ] Build a behavioral baseline before relying on anomaly alerts
  Without a baseline, anomaly detection generates excessive false positives.
  mon-005

Network & Infrastructure

HIGH 5 controls

- [ ] Do not expose MCP servers directly to the public internet
  Public exposure dramatically increases attack surface.
  net-001
- [ ] Place MCP servers behind a gateway or reverse proxy
  Centralizes TLS termination, auth enforcement, and rate limiting.
  net-002
- [ ] Apply egress filtering to MCP servers
  Prevents arbitrary outbound connections and exfiltration.
  net-003
- [ ] Run MCP server processes as a non-root, dedicated service account
  Limits blast radius of OS-level compromise.
  net-004
- [ ] Keep MCP server dependencies up to date
  Supply chain vulnerabilities in libraries are a real and exploited risk.
  net-005