What is AI agent sandboxing?

AI agent sandboxing means running an AI agent and the tools it uses inside a constrained execution environment that limits its access to the file system, network, credentials and system calls. It matters because agents execute generated code, call tools across trust domains and handle sensitive data, and any of them can be hijacked through prompt injection. Common layers are microVMs or gVisor-style kernels, container security profiles (seccomp, AppArmor, SELinux), read-only and ephemeral file systems, egress allowlists and secrets kept outside the sandbox.

HOW IT WORKS

01 Why agents need it more than ordinary services

A normal web service runs code its developers wrote and reviewed. An agent runs code and commands that a model decided on at runtime, influenced by every document, web page and tool result it read along the way. That changes the threat model:

- indirect prompt injection can turn "summarise this repository" into "run this install script".
- Generated code can be wrong in harmful ways even without an attacker: deleting files, looping on a paid API, or writing secrets into logs.
- Tools span trust domains, so misuse of one tool (a file reader) can feed another (an HTTP client) and exfiltrate data. See LLM data exfiltration.
- Multi-tenant platforms run many users' agents on shared infrastructure, where a container escape becomes a cross-customer breach.

02 The layers of a good agent sandbox

- Isolation boundary. Prefer microVMs or user-space kernels such as gVisor for untrusted code; a plain container shares the host kernel, and a a container escape turns one bad tool call into host access.
- System call and capability limits. seccomp, AppArmor or SELinux profiles; no privileged mode, no extra Linux capabilities, non-root user.
- File system policy. Read-only root, an ephemeral scratch directory, and only the specific project paths the task needs. Never mount the home directory, the Docker socket or credential files.

SOURCES

[1] MCP Security Best Practices

[2] Model Context Protocol specification

[3] OWASP Top 10 for Agentic Applications (2026)

[4] MITRE ATLAS

- Network egress control. Default deny outbound; allowlist the few hosts the task needs, and route tool traffic through an an MCP gateway or egress proxy that logs it. Block cloud metadata addresses.
- Secrets outside the box. The sandbox should never hold long-lived credentials. Inject short-lived, narrowly scoped tokens per task, or have the gateway attach them outside the sandbox.
- Resource limits and lifetime. CPU, memory, process count and wall-clock limits, and a fresh sandbox per task or session so nothing persists for the next one.

03 How to test a sandbox

Assume the model is fully compromised and try to get out. From inside the sandbox, test whether you can read host files or other tenants' data, reach the cloud metadata service or internal networks, find credentials in environment variables or mounted files, persist anything between sessions, or exhaust shared resources. Then test the path in: whether indirect prompt injection in a document the agent reads can make it run code at all, and whether that code runs inside the sandbox or somewhere the designers forgot.

Test your AI application before an attacker does.
securelayer7.net/learn/ai-security/ai-agent-sandboxing

Open online