Credential-store reads

Flags an AI agent reading a file that holds credentials into its context — a `.env`, `~/.aws/credentials`, an MCP configuration with tokens in it, a private SSH key. **Default: warn.**

Why this is separate from secret scanning

Secret scanning catches a key when its *shape* shows up in the traffic. By then the value is already in the model's context. This detector catches the *read itself*: the moment an agent opens the store, whether or not a recognizable key ever appears in what it gets back. It is also what infostealers and misbehaving agents went after first in 2026.

What it looks at

Only two things, both in the agent's own tool calls: the file path a file tool (Read, Edit, Write and their equivalents) was asked to open, and the shell command a terminal tool ran (`cat .env`, `source ~/.aws/credentials`, input redirected from a store). A mention of `.env` in a message is not a read and is ignored. Listing a directory or checking that a file exists is not a read either.

It recognizes a fixed list of stores: `.env` files (but not `.env.example`, `.env.sample` or `.env.template`), Claude Code configuration, MCP configuration files, git credentials, AWS credentials, gcloud credentials, Kubernetes config, `.npmrc`, Docker config, private SSH keys and `.netrc`. The list is in the product, not in your settings — nobody can turn this into a tool for listing files.

What is recorded

The **category** of store and a count — for example `aws_credentials × 1`. Never the path, never the contents. Your developers' directory layout does not end up in the audit record.

Setting it

Warn shows the read in Activity and to the developer. **Block** refuses the request that would carry the file's contents to the model — the honest version of "keep `.env` out of the model's context", which secret scanning alone cannot do. Expect a developer whose agent legitimately needs a local `.env` to notice; the message names

the category so they know why.

When a read is not blocked, it also counts as the *private data* part of the lethal-trifecta check for that device, so untrusted content plus an outbound call afterwards is caught even though the data came through a built-in tool rather than an MCP server.

Revision #1
Created 2026-09-29 14:39:34 UTC by Sentilai Docs
Updated 2026-09-29 14:39:35 UTC by Sentilai Docs