

Чек-лист: диагностика Claude Code plugin с MCP

Используйте пункты последовательно и фиксируйте результат в той среде, где возникает проблема. Не заменяйте проверку MCP-сессии результатом из другого CLI-процесса.

1. Проверить пакет plugin

- Запустите `claude plugin validate /absolute/path/to/plugin`.
- Проверьте `.claude-plugin/plugin.json`, структуру `skills/` и наличие MCP-конфигурации.
- Для локального MCP не используйте жёстко заданный путь машины разработчика: применяйте `${CLAUDE_PLUGIN_ROOT}` там, где это поддерживается.
- Запустите Claude Code с `--debug` и сохраните диагностический вывод без секретов.

2. Разделить состояния

- Зафиксируйте отдельно `discovery` файлов `plugin` и регистрацию `Skill`.
- Проверьте, принята ли MCP-конфигурация и подготовлен ли запуск или удалённое соединение.
- Проверьте подключение сервера, авторизацию, `trust state` и `approval`.
- Подтвердите `tool exposure` безопасным вызовом известного `read-only` инструмента.

3. Выполнить A/B-тест

- В варианте A используйте установленный `enabled plugin`.
- В варианте B передайте тот же каталог через абсолютный `--plugin-dir`.
- Не меняйте `prompt`, рабочую папку, версию Claude Code, переменные окружения и учётные данные.
- Сравните `/mcp`, `debug-вывод`, состояние сервера и результат вызова ожидаемого `tool`.

4. Убрать неоднозначность диагностики

- Для чистого прогона `cache` используйте `MCP_DISCOVERY_CACHE=0`.
- Если нужно исключить отложенную загрузку схем, используйте `ENABLE_TOOL_SEARCH=false`.
- Для безинтерактивного прогона используйте `--output-format stream-json` и проверьте `mcp_server_errors` в `system/init`.
- На корпоративном устройстве зафиксируйте `managed settings`, включая политики `sideloading`.