Ай-ТИ Фреш

ТЕХНИЧЕСКИЙ РАЗБОР

Claude Code как платформа: плагины, скиллы, хуки, MCP

Наша архитектура расширений: корпоративный плагин, MCP-интеграции и контур контроля hooks

Июль 2026

itfresh.ru · ИТ-аутсорсинг для юридических лиц

О ЧЁМ ЭТОТ РАЗБОР

Суть проблемы

Команды начинают пользоваться Claude Code стихийно: каждый инженер ставит расширения из открытых репозиториев, конфиги живут на личных машинах, чужие плагины получают доступ к файлам и ключам. Мы переводим это в управляемую платформу: корпоративный плагин, приватный marketplace, контур permissions и hooks — иначе утечка данных клиента лишь вопрос времени.

Почему это важно бизнесу

- Чужой плагин исполняет код с правами сотрудника: один непроверенный hook — и ключи или базы клиентов уходят наружу
- Без единого стандарта каждый инженер настраивает ИИ по-своему: результат нестабилен, передача задач буксует
- Рутинные связки CRM–1С–почта съедают часы инженеров, хотя закрываются готовыми MCP-подключениями
- Настройки на личных машинах не версионируются: уход сотрудника означает потерю отлаженных процессов

ЧТО МЫ ЗАКЛАДЫВАЕМ

Ключевые параметры реализации

2.x

мажорная ветка Claude Code CLI: plugins, skills, hooks и MCP — штатные механизмы ядра

code.claude.com/docs

30+

событий хуков — от PreToolUse до SessionEnd: точка политики на каждом шаге агента

по докам hooks

3

scope конфигурации MCP (local / project / user): разводим личные и командные подключения

по докам MCP

exit 2

код выхода хука = блокировка: PreToolUse гасит вызов инструмента даже при allow

по докам hooks

600 с

дефолтный таймаут command-хука; свои проверки укладываем в явный timeout 5–30 с

по докам hooks

1024

лимит символов description в SKILL.md — от его формулировки зависит автоподхват навыка моделью

по докам skills

РЕАЛИЗАЦИЯ 1 ИЗ 3

Единица поставки: корпоративный плагин

Что настраиваем

Репозитории клиентов + рабочие места инженеров ITfresh; один плагин itfresh-core на всю команду

Как мы это делаем

1. Каркас: манифест .claude-plugin/plugin.json (name, version по semver, description); каталоги skills/, agents/, hooks/ (hooks.json) и .mcp.json — строго в корне плаг…
2. Скиллы: skills/<имя>/SKILL.md с триггерами в description; вызов по неймспейсу /itfresh-core:deploy, аргументы через $ARGUMENTS
3. Дистрибуция: приватный git-репозиторий как marketplace (marketplace.json); у инженеров /plugin marketplace add и /plugin install
4. Релизы: claude plugin validate в CI, bump поля version; у инженеров claude plugin update, в живой сессии /reload-plugins подхватывает без перезапуска
5. Отладка до публикации: claude --plugin-dir ./itfresh-core грузит рабочую копию поверх установленной

РЕЗУЛЬТАТ

Один версионируемый артефакт вместо россыпи личных конфигов: у всех инженеров одинаковые навыки, хуки и подключения; обновление — bump версии в git, новый сотрудник получает отлаженное окружение одной командой установки.

КЛЮЧЕВОЙ НЮАНС

Внутри .claude-plugin/ лежит только plugin.json — компоненты кладём в корень плагина, иначе они молча не загрузятся; плагинные скиллы всегда неймспейсятся (/plugin:skill) и не конфликтуют с локальными.

РЕАЛИЗАЦИЯ 2 ИЗ 3

Интеграции: MCP-серверы вместо ручной склейки

Что настраиваем

Контур клиента: 1С через OData, CRM, почтовый сервер, SSH-парк — подключаем к агенту штатным протоколом MCP

Как мы это делаем

1. Подключаем серверы через claude mcp add с явным scope: user — личные, project — общий .mcp.json в git-репозитории клиента
2. Транспорт по месту: stdio для локальных обвязок (1С OData, SSH-парк), http (streamable-http) с OAuth для внешних API; SSE не берём — deprecated
3. Секреты только через переменные окружения — в .mcp.json подстановка ${VAR} и ${VAR:-default}, сам файл коммитится без ключей
4. Лимиты: MCP_TIMEOUT (в мс) на старт сервера, MAX_MCP_OUTPUT_TOKENS против распухания контекста ответами
5. В permissions — точечный allowlist инструментов вида mcp__server__tool вместо wildcard на весь сервер

РЕЗУЛЬТАТ

Агент связывает CRM, 1С, почту и мониторинг штатными инструментами: сценарий «заявка → задача → уведомление» собирается декларативно, без самописных ботов и cron-скриптов, и переносится между проектами копированием .mcp.json.

КЛЮЧЕВОЙ НЮАНС

MCP-инструмент — внешний код с правами сессии: разрешаем конкретные инструменты, а не сервер целиком; большие выборки пагинируем на стороне сервера, иначе один ответ вытесняет из контекста саму задачу.

РЕАЛИЗАЦИЯ 3 ИЗ 3

Контур контроля: hooks, permissions, субагенты

Что настраиваем

Боевые репозитории и серверы клиентов — всё, где агент касается продовых данных

Как мы это делаем

1. PreToolUse-хук с matcher Bash: скрипт читает JSON со stdin (jq .tool_input.command), опасные паттерны гасит exit 2 — вызов блокируется
2. PostToolUse на Edit|Write: автолинт и форматирование изменённых файлов сразу после правки агента
3. permissions.deny: Read(./.env), Read(./secrets/**) — агент физически не читает ключи; режимы plan/acceptEdits под тип задачи
4. Субагенты в .claude/agents/*.md: ревьюеру только Read+Grep, дешёвые проверки — на модель haiku через поле model
5. Managed settings организации: обязательные политики поверх пользовательских, запрет disableAllHooks

РЕЗУЛЬТАТ

Защита боевых данных работает на уровне механики, а не договорённостей: запрещённые команды не выполняются, секреты не попадают в контекст модели, каждый шаг агента проходит через журналируемые точки контроля.

КЛЮЧЕВОЙ НЮАНС

exit 2 блокирует действие даже если JSON-вывод хука содержит permissionDecision: allow; дефолтный таймаут command-хука 600 с — быстрым проверкам ставим свой timeout, чтобы зависший скрипт не держал сессию.

ЧТО МЫ ВЫУЧИЛИ НА ПРАКТИКЕ

Подводные камни

- ✕ **Компоненты внутри .claude-plugin/**
  Там живёт только plugin.json; skills/, agents/, hooks/ кладём в корень плагина — иначе компоненты молча не загружаются
- ✕ **Секреты в .mcp.json под git**
  Project-scope файл коммитится в репозиторий; ключи выносим в env и подставляем через ${VAR} — в файле остаются лишь имена переменных
- ✕ **Сторонний плагин без ревью**
  Hooks, MCP-серверы и bin/ исполняются с правами сотрудника; ставим только после чтения кода, claude plugin validate и пиннинга версии
- ✕ **Размытый description скилла**
  Модель подхватывает скилл по description (до 1024 симв.); пишем конкретные триггеры и проверяем автовызов на тестовых промптах
- ✕ **Хук без явного timeout**
  Дефолт command-хука — 600 с: зависший скрипт держит сессию; задаём timeout в секундах на каждый handler
- ✕ **Один агент на все задачи**
  Контекст распухает, качество падает; поиск и ревью выносим в субагентов с урезанным списком tools и подходящей моделью
- ✕ **Обновления плагина не доехали**
  Без bump поля version в plugin.json marketplace не отдаст новую версию; релиз = semver-бамп + claude plugin update у инженеров (/reload-plugins — в ж…
- ✕ **MCP-ответы съедают контекст**
  Большая выборка одним ответом вытесняет задачу; ограничиваем MAX_MCP_OUTPUT_TOKENS и строим пагинацию в самом сервере

УРОВНИ РЕАЛИЗАЦИИ

Как правильно

МИНИМУМ

- CLAUDE.md и .claude/skills в git проекта — база знаний агента версионируется
- permissions: deny на чтение .env и секретов, allow на типовые команды
- .mcp.json в проекте, ключи только через переменные окружения

НОРМАЛЬНО

- Корпоративный плагин: skills + agents + hooks одним артефактом с semver-версией
- Приватный marketplace в git; установка /plugin install, обновление claude plugin upd…
- PreToolUse/PostToolUse-хуки: блок опасных команд, автолинт после правок
- Субагенты с урезанным списком tools под ревью, поиск и деплой

ХОРОШО

- Managed settings: политика организации поверх личных, запрет disableAllHooks
- CI-контроль: claude plugin validate --strict на каждый релиз плагина
- Собственные MCP-серверы к 1С/CRM/почте вместо самописных скриптов-обвязок
- Регламент ревью сторонних расширений: код, права, пиннинг коммита

ПРОВЕРЬТЕ У СЕБЯ

Чек-лист самопроверки

- [ ] Конфигурация Claude Code (.claude/, .mcp.json, плагины) хранится в git, а не на личных машинах?
- [ ] Каждое стороннее расширение прошло ревью кода перед установкой (hooks, MCP, bin/)?
- [ ] Секреты закрыты: deny на Read(./.env), ключи MCP-серверов только в переменных окружения?
- [ ] Есть PreToolUse-хук, блокирующий опасные команды кодом выхода 2?
- [ ] Версии плагинов зафиксированы по semver, процесс обновления описан?
- [ ] Субагенты ограничены списком tools и моделью под тип задачи?
- [ ] Прописаны таймауты хуков и лимит MAX_MCP_OUTPUT_TOKENS?
- [ ] Отключение хуков (disableAllHooks) запрещено на уровне managed settings организации?
- [ ] Команда знает, какие данные нельзя отправлять в модель (ПДн клиентов, коммерческая тайна)?

Если хотя бы на два вопроса ответ «нет» или «не знаю» — тема требует внимания.

ЧТО МЫ ДЕЛАЕМ

Как поможет ITFresh

ITFresh — ИТ-аутсорсинг для юридических лиц до 50 рабочих мест в Москве и области. 15+ лет практики, собственная инфраструктура в дата-центре МТС (8 серверов Dell Xeon Platinum).

- Аудит текущего использования Claude Code и сборка корпоративного плагина под ваши процессы
- Приватный plugin-marketplace в вашем git: версионирование, ревью, установка одной командой
- MCP-подключения к вашим системам: 1С (OData), CRM, почта, SSH-парк — с изоляцией секретов
- Контур безопасности: permissions, PreToolUse-хуки, managed settings, регламент ревью расширений
- Обучение инженеров и передача регламента сопровождения платформы

15+
лет в ИТ-поддержке

50
рабочих мест — наш профиль

МТС
дата-центр, Москва

КОНТАКТЫ

Обсудить вашу задачу

Сайт	**itfresh.ru**
Телефон	**+7 903 729-62-41**
Telegram	**@ITfresh_Boss**

Бесплатно посмотрим вашу инфраструктуру по этому чек-листу и скажем, где тонко — без обязательств.

itfresh.ru

ДОКУМЕНТАЦИЯ И НАШ СТЕК

Техническая база

01 Create plugins / Plugins reference (plugin.json) (code.claude.com — 2026)

02 Hooks reference (события, exit-коды, таймауты) (code.claude.com — 2026)

03 Agent Skills (SKILL.md, автоподхват навыков) (code.claude.com — 2026)

04 MCP — Model Context Protocol (scopes, транспорты) (code.claude.com — 2026)

05 Subagents (.claude/agents, tools, model) (code.claude.com — 2026)

06 Settings и permissions (allow/deny, managed policy) (code.claude.com — 2026)

07 Plugin marketplaces (приватная дистрибуция) (code.claude.com — 2026)

08 Шаблон ITfresh: корпоративный плагин + чек-лист ревью (itfresh.ru — наш)

Основано на официальной документации продуктов и нашей практике внедрения.