LLM Remote

Puente Cifrado Telegram ↔ IA Multi-Proveedor

REDEGAL - A SMART COMPANY

Versión 1.3 · Febrero 2026 · Documento interno

Índice

1. Introducción

LLM Remote es una herramienta interna que permite a los miembros del equipo de Redegal interactuar con múltiples proveedores de IA (Claude Code, OpenAI, Gemini, Groq) directamente desde Telegram, con cifrado de nivel bancario.

¿Para qué sirve?

- **Desarrollo remoto** — Ejecuta Claude Code en tu máquina desde cualquier lugar
- **Consultas rápidas** — Pregunta a GPT-4o, Gemini o Llama desde el móvil
- **Multi-proyecto** — Cambia entre proyectos con un comando
- **Seguro** — Todo cifrado, autenticado y con audit log

> **Requisitos previos**
>
> Node.js 20+, un bot de Telegram (de @BotFather), y opcionalmente Claude Code CLI.

2. Proveedores IA

LLM Remote soporta 5 proveedores. Cambia entre ellos con `/ia <nombre>` en Telegram.

Proveedor	Comando	Modo	Coste
🟣 **Claude Code**	`/ia claude`	**Agentic** Lee/escribe ficheros, ejecuta comandos	**Según plan**
🟢 **OpenAI GPT-4o**	`/ia openai`	Chat API	**Pay-per-use**
🔵 **Gemini 2.5 Flash**	`/ia gemini`	Chat API	**Gratis** 20 req/día
🟠 **Groq Llama 3.3**	`/ia groq`	Chat API (ultra-rápido)	**Gratis** 30 req/min
🟣 **Anthropic Sonnet**	`/ia anthropic`	Chat API (sin agentic)	**Pay-per-use**

> **Recomendación para uso diario**
>
> Usa **Groq** para consultas rápidas (gratis, respuesta en <1s) y **Claude Code** cuando necesites que la IA modifique archivos o ejecute comandos en tu máquina.

Diferencia entre Claude Code y Anthropic API

- **Claude Code** (`/ia claude`) — Modo agentic completo. Puede leer ficheros, escribir código, ejecutar comandos, navegar tu proyecto. Usa el CLI de Claude Code.
- **Anthropic API** (`/ia anthropic`) — Solo chat. Responde preguntas pero no tiene acceso a tu sistema de archivos. Usa la API directamente.

3. Instalación

3.1 Instalación automática (recomendado)

Un solo comando que verifica requisitos, descarga, instala y configura:

```
bash installer.sh
```

El instalador hace todo automáticamente:

1. Verifica Node.js 20+, npm, git, Claude Code CLI
2. Descarga o actualiza el proyecto
3. Instala dependencias npm
4. Lanza el configurador interactivo (6 pasos)
5. Opcionalmente crea un servicio de auto-arranque

3.2 Instalación manual

```
# 1. Clonar repositorio
git clone <repo-url> ~/llm-remote
cd ~/llm-remote

# 2. Instalar dependencias
npm install

# 3. Configurar
npm run setup

# 4. Arrancar
npm start
```

3.3 Requisitos

Requisito	Versión	Obligatorio	Notas
Node.js	20+	Sí	`brew install node` o nodejs.org

npm	cualquiera	Sí	Viene con Node.js
git	cualquiera	Sí	`xcode-select --install` (macOS)
Claude Code CLI	cualquiera	No	`npm i -g @anthropic-ai/claude-code`

> **Sin Claude Code CLI**
>
> Si no instalas Claude Code CLI, el proveedor `/ia claude` no funcionará. Los demás proveedores (OpenAI, Gemini, Groq, Anthropic) sí funcionarán sin problema.

4. Configuración

4.1 Crear bot de Telegram

1. Abre **@BotFather** en Telegram
2. Envía `/newbot`
3. Elige un nombre (ej: *"Mi LLM Remote"*)
4. Elige un username (ej: *"mi_llm_remote_bot"*)
5. Copia el **token** que te da BotFather

4.2 Obtener tu Telegram User ID

1. Abre **@userinfobot** en Telegram
2. Envía `/myid`
3. Copia el número (ej: *123456789*)

4.3 Obtener API keys (opcional)

Proveedor	URL	Coste
Gemini	https://aistudio.google.com/apikey	Gratis
Groq	https://console.groq.com/keys	Gratis
OpenAI	https://platform.openai.com/api-keys	De pago
Anthropic	https://console.anthropic.com/settings/keys	De pago

4.4 Wizard de configuración

El wizard (`npm run setup`) te guía en 6 pasos:

```
  Token del bot: <pegar token>
  IDs autorizados: <tu user ID>

── 2/6  Seguridad ──
  PIN: <auto-generado o personalizado>
  Contraseña maestra: <auto-generada>

── 3/6  Sesión y Límites ──
  Timeout: 15 min
  Max comandos/min: 10

── 4/6  Claude Code CLI ──
  Binario: claude
  Directorio: /Users/tu-usuario

── 5/6  Proveedores IA ──
  OpenAI, Gemini, Anthropic, Groq (opcionales)

── 6/6  Logging ──
  Nivel: info
```

Reconfigurar en cualquier momento

Ejecuta `npm run setup` de nuevo. Los valores actuales aparecen como defaults.

5. Uso desde Telegram

5.1 Primer uso

1. Arranca el bot: `npm start` (o el servicio lo hace automáticamente)
2. Abre tu bot en Telegram
3. Envía `/start` para ver los comandos
4. Autentícate: `/auth <tu-PIN>`
5. El mensaje del PIN se borra automáticamente por seguridad
6. Escribe cualquier mensaje — va directamente al proveedor IA activo

5.2 Cambiar proveedor IA

```
# Ver proveedores disponibles
/ia

# Cambiar a Groq (gratis, ultra-rápido)
/ia groq

# Cambiar a Claude Code (agentic)
/ia claude

# Cambiar a Gemini (gratis)
/ia gemini
```

5.3 Trabajar con proyectos

```
# Ver directorio actual
/project

# Cambiar a otro proyecto
/project ~/mi-proyecto

# Ahora los comandos a Claude Code operan en ~/mi-proyecto
Refactoriza el archivo main.js
```

5.4 Sesiones y seguridad

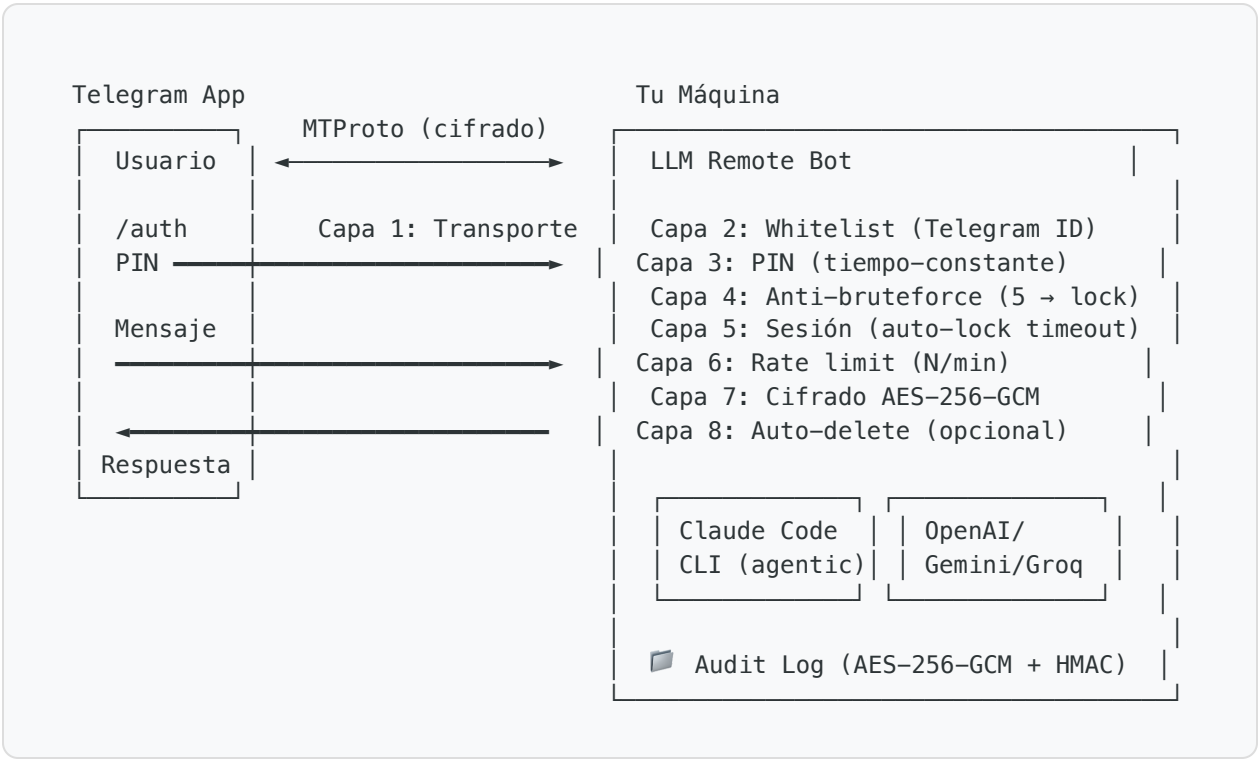
```
# Ver estado de la sesión
/status

# Bloquear manualmente
/lock

# Re-autenticarse
/auth <PIN>

# Ver historial (descifrado)
/history
```

6. Referencia de Comandos

Comando	Descripción	Ejemplo
`/start`	Muestra ayuda y proveedores disponibles	`/start`
`/auth <PIN>`	Autenticarse. El mensaje se borra automáticamente.	`/auth <tu-PIN>`
`/ask <prompt>`	Enviar prompt explícito al proveedor activo	`/ask explica qué hace este regex`
`/ia [nombre]`	Ver o cambiar proveedor IA	`/ia groq`
`/project [ruta]`	Ver o cambiar directorio de trabajo	`/project ~/app`
`/status`	Info de sesión, proveedor y proceso	`/status`
`/history`	Últimos 15 comandos (descifrado del audit log)	`/history`
`/kill`	Matar proceso Claude Code en ejecución	`/kill`
`/lock`	Bloquear sesión inmediatamente	`/lock`
`/help`	Mostrar todos los comandos	`/help`
(texto libre)	Se envía directamente al proveedor activo	*¿Cómo optimizo esta query SQL?*

7. Arquitectura de Seguridad

7.1 Modelo de 8 capas



7.2 Cifrado

Componente	Especificación
Algoritmo	AES-256-GCM (cifrado autenticado)
Derivación de clave	PBKDF2 con 310.000 iteraciones + SHA-512
IV (vector de inicialización)	16 bytes aleatorios por mensaje
Salt	32 bytes aleatorios por mensaje
Integridad	HMAC-SHA256 sobre todo el payload
Resultado	Cada cifrado es único, incluso con el mismo texto

7.3 Autenticación

- **Whitelist** — Solo los Telegram IDs configurados pueden interactuar. Usuarios no autorizados son ignorados silenciosamente.
- **PIN** — Comparación en tiempo constante para prevenir ataques de timing.
- **Anti-bruteforce** — 5 intentos fallidos → bloqueo de 15 minutos.
- **Sesiones** — Auto-lock tras inactividad configurable (default: 15 min).
- **Auto-delete** — El mensaje con el PIN se borra automáticamente de Telegram.

> **Importante: Seguridad por instalación**
>
> Cada instalación genera su propia contraseña maestra de cifrado. Las claves no se comparten entre usuarios. Si pierdes tu `.env`, no hay forma de recuperar los datos del audit log.

8. Arquitectura Técnica

8.1 Stack tecnológico

Componente	Tecnología	Justificación
Runtime	Node.js 20+	Async/await nativo, fetch nativo, sin dependencias nativas
Bot Telegram	grammY	Framework moderno, TypeScript-ready, middleware
Cifrado	node:crypto	Módulo nativo, AES-256-GCM, PBKDF2, HMAC
APIs IA	fetch nativo	Zero dependencias, HTTP estándar
Configuración	dotenv	Estándar de la industria
Audit log	NDJSON cifrado	Append-only, sin BD, portátil

8.2 Estructura del proyecto

```
llm-remote/
├── src/
│   ├── index.js              # Punto de entrada
│   ├── bot.js                # Bot Telegram + handlers
│   ├── setup.js              # Configurador interactivo
│   ├── auth/
│   │   ├── guard.js          # Whitelist + anti-bruteforce
│   │   └── session.js        # Sesiones + timeout
│   ├── crypto/
│   │   └── cipher.js         # AES-256-GCM + HMAC + PBKDF2
│   ├── providers/
│   │   ├── base.js           # Interfaz base
│   │   ├── manager.js        # Gestor multi-proveedor
│   │   ├── claude.js         # Claude Code CLI
│   │   ├── openai.js         # OpenAI GPT-4o
│   │   ├── gemini.js         # Google Gemini
│   │   ├── groq.js           # Groq Llama 3.3
│   │   └── anthropic.js      # Anthropic Sonnet
│   ├── claude/
│   │   └── formatter.js      # Chunking para Telegram
│   ├── security/
```

```
│   │   ├── ratelimit.js      # Rate limiting
│   │   └── audit.js          # Audit log cifrado
│   └── utils/
│       ├── config.js         # Configuración
│       ├── logger.js         # Logger
│       └── keygen.js         # Generador de claves
├── tests/
│   └── crypto.test.js        # 11 tests de cifrado
├── installer.sh              # Instalador corporativo
├── .env.example              # Plantilla de config
└── package.json
```

8.3 Dependencias

Solo **2 dependencias** de producción (zero nativas):

Paquete	Versión	Función
grammy	^1.31	Framework bot Telegram
dotenv	^16.4	Variables de entorno

9. Resolución de Problemas

Problema	Causa	Solución
El bot no responde	No está corriendo o token incorrecto	Verificar `npm start` y el token en `.env`
"Session expired"	Timeout de inactividad	Enviar `/auth <PIN>` de nuevo
"Locked out"	5 intentos de PIN fallidos	Esperar 15 minutos
"Rate limited"	Demasiados comandos por minuto	Esperar unos segundos
"Provider not configured"	Falta API key del proveedor	Ejecutar `npm run setup` y añadir la API key
Claude Code no funciona	CLI no instalado	`npm i -g @anthropic-ai/claude-code`
Gemini error 429	Límite diario alcanzado (20 req)	Cambiar a Groq: `/ia groq`
Error de cifrado	.env corrupto o cambiado	El audit log anterior no será legible. Borrar `data/` y empezar de nuevo.

Logs

```
# Ver logs en tiempo real
tail -f ~/llm-remote/data/llm-remote.log

# Si usa servicio en macOS
tail -f ~/llm-remote/data/llm-remote.log

# Si usa servicio en Linux
journalctl --user -u llm-remote -f
```

Reconfigurar

```
cd ~/llm-remote && npm run setup
```

Desinstalar

```
bash ~/llm-remote/installer.sh --uninstall
```

10. Variables de Entorno

Variable	Req.	Default	Descripción
`TELEGRAM_BOT_TOKEN`	Sí	—	Token del bot de Telegram
`AUTHORIZED_USERS`	Sí	—	IDs Telegram autorizados (coma)
`AUTH_PIN`	Sí	—	PIN de autenticación
`MASTER_PASSWORD`	Sí	—	Contraseña maestra cifrado (16+ chars)
`SESSION_TIMEOUT_MIN`	No	15	Timeout sesión (minutos)
`RATE_LIMIT_PER_MIN`	No	10	Max comandos/minuto
`AUTO_DELETE_SEC`	No	0	Auto-borrar mensajes (0 = off)
`CLAUDE_BIN`	No	claude	Ruta binario Claude Code
`DEFAULT_WORK_DIR`	No	$HOME	Directorio trabajo por defecto
`MAX_CONCURRENT`	No	2	Procesos Claude simultáneos
`OPENAI_API_KEY`	No	—	API key OpenAI
`OPENAI_MODEL`	No	gpt-4o	Modelo OpenAI
`GEMINI_API_KEY`	No	—	API key Google Gemini
`GEMINI_MODEL`	No	gemini-2.5-flash	Modelo Gemini
`ANTHROPIC_API_KEY`	No	—	API key Anthropic
`ANTHROPIC_MODEL`	No	claude-sonnet-4	Modelo Anthropic
`GROQ_API_KEY`	No	—	API key Groq
`GROQ_MODEL`	No	llama-3.3-70b	Modelo Groq
`LOG_LEVEL`	No	info	debug/info/warn/error

LLM Remote v1.3 · Redegal - A Smart Company · Febrero 2026