OpenClaw en Linux

Guía completa de setup en Ubuntu 24

Agentes autónomos · Telegram · OpenRouter · Web Search

Luis Miguel Triana Rueda

Software Engineer · AI Builder · Socorro, Santander

Contenido

1 ¿Qué es OpenClaw?

OpenClaw es un framework de agentes autónomos de IA que permite configurar y ejecutar asistentes inteligentes directamente en tu propio servidor. Esta guía documenta el proceso completo de setup en Ubuntu 24.

La configuración cubre: integración con **Telegram**, múltiples modelos de lenguaje vía **OpenRouter**, búsqueda web con **Gemini**, y personalización completa de la identidad del agente.

2 Preparación del sistema

OpenClaw requiere **Node.js 22+**. Usamos nvm para gestionar versiones sin afectar proyectos existentes.

```
# Instalar Node.js 22 con nvm
nvm install 22
nvm use 22
nvm alias default 22
node --version  # v22.22.1

# Proteger proyectos existentes (ej. Next.js)
echo "20" > /ruta/tu-proyecto/.nvmrc
```

■ **Tip:** Agregar un archivo .nvmrc en proyectos existentes previene conflictos de versión al cambiar entre directorios.

3 Instalación

La instalación se realiza con el script oficial de OpenClaw:

```
curl -fsSL https://openclaw.ai/install.sh | bash
```

Opciones del Onboarding:

Opción	Valor recomendado
Seguridad	Yes
Modo	QuickStart
Provider	OpenRouter
Canal	Usuario de Telegram

Gateway	localhost 127.0.0.1:18789

4 Configuración de modelos

Los modelos se configuran vía OpenRouter, lo que permite acceder a múltiples proveedores con una sola API key.

```
# Configurar modelo por defecto
openclaw config set agents.defaults.model openrouter/deepseek/deepseek-v3.2
openclaw gateway restart

# Configurar API key
nano ~/.openclaw/agents/main/agent/auth-profiles.json
```

Modelos en openclaw.json:

```
"models": {
  "openrouter/auto":                       { "alias": "OpenRouter"       },
  "openrouter/deepseek/deepseek-v3.2":     { "alias": "DeepSeek V3.2"    },
  "openrouter/deepseek/deepseek-r1":       { "alias": "DeepSeek R1"      },
  "openrouter/google/gemini-2.0-flash-001": { "alias": "Gemini 2.0 Flash" },
  "openrouter/qwen/qwen3-235b-a22b":       { "alias": "Qwen3 Agentes"    }
}
```

5 Seguridad — Allowlist Telegram

Para proteger el bot de accesos no autorizados, se configura una allowlist con el ID del usuario permitido.

```
# Configurar allowlist (solo tu user ID)
openclaw config set channels.telegram.allowFrom '["TU_USER_ID"]'
openclaw gateway restart

# Aprobar el pairing inicial
openclaw pairing approve telegram CODIGO_DE_PAIRING
```

■■ **Importante: Nunca compartir el código de pairing ni tu User ID de Telegram públicamente.**

6 Web Search — Gemini Google Search

Configuración de búsqueda web en tiempo real usando la API de Gemini. Obtén tu API key en **aistudio.google.com**.

```
# Opción 1: Wizard interactivo
openclaw configure --section tools
# Navegar: Web tools > Enable web_search > Yes > Gemini > pegar API key

# Opción 2: Comandos directos
openclaw config set tools.web.search.provider gemini
openclaw config set tools.web.search.gemini.apiKey TU_API_KEY
openclaw gateway restart
```

7 Workspace del agente

El workspace es el directorio donde el agente almacena su configuración, contexto e identidad.

```
# Crear directorio del agente
mkdir ~/mi-agente

# Asignar como workspace activo
openclaw config set agents.defaults.workspace ~/mi-agente
openclaw gateway restart
```

8 Identidad del agente

La personalidad del agente se define a través de tres archivos markdown que controlan su nombre, conocimiento del usuario y comportamiento.

IDENTITY.md — Nombre y tema del agente

```
cat > ~/mi-agente/IDENTITY.md <<'EOF'
# Identity
Name: [Nombre del agente]
Theme: [Tema o personalidad]
Emoji: [Emoji representativo]
EOF
```

USER.md — Contexto del usuario

```
cat > ~/mi-agente/USER.md <<'EOF'
# About the User
- [Rol o profesion]
- [Ubicacion]

# Active Projects
- [Proyecto 1]: [Descripcion breve]

# Technical Stack
- Languages: [Lenguajes]
- Frameworks: [Frameworks]
- AI/ML: [Herramientas de IA]
EOF
```

SOUL.md — Comportamiento y valores

```
cat > ~/mi-agente/SOUL.md <<'EOF'
# Soul

## Core Behavior
- Always confirm before executing any action
- Never act autonomously without explicit approval

## MANDATORY RULE — NO EXCEPTIONS
Before ANY file creation, modification or command execution:
  1. Describe exactly what you are about to do
  2. Ask for confirmation
  3. Wait for approval
  4. Only then execute
EOF

openclaw gateway restart
```

9 Stack final

Rol	Modelo	Estado
Default / Chat	openrouter/deepseek/deepseek-v3.2	■ Activo
Razonamiento	openrouter/deepseek/deepseek-r1	■ Activo
Contexto largo	openrouter/google/gemini-2.0-flash-001	■ Activo
Agentes / Coding	openrouter/qwen/qwen3-235b-a22b	■ Activo
Web Search	Gemini Google Search	■ Activo
Identidad	Custom (IDENTITY + USER + SOUL)	■ Activo

Cambiar modelo desde Telegram:

```
/model DeepSeek R1
/model Gemini 2.0 Flash
/model Qwen3 Agentes
/model DeepSeek V3.2
```

10 Comandos útiles

Terminal:

```
openclaw gateway status    # ver estado del gateway
openclaw gateway restart   # reiniciar el gateway
openclaw logs --follow     # ver logs en tiempo real
openclaw models list       # listar modelos disponibles
```

Comandos desde Telegram:

Comando	Acción
/usage	Ver gasto acumulado en OpenRouter
/new	Iniciar nueva sesión de chat
/reset	Resetear la sesión actual
/stop	Detener una respuesta larga
/compact	Comprimir el contexto de la conversación

/think	Activar razonamiento profundo

11 Infraestructura

Componente	Detalle
Sistema operativo	Ubuntu 24.04 LTS
Gateway local	localhost:127.0.0.1:18789
Web UI	http://127.0.0.1:18789
Canal de acceso	Bot de Telegram personal
Workspace	~/mi-agente
API provider	OpenRouter ($10 USD recargados)
Gasto acumulado	~$0.079 USD
Claude Code	Instalado y configurado

12 Bug conocido — Issue #48665

■■ Importante: El Control UI de OpenClaw envía el ID del modelo sin el prefijo del provider, causando errores al cambiar modelos desde la interfaz web.

Workaround — usar el prefijo completo desde Telegram:

```
/model openrouter/deepseek/deepseek-r1
```

Fix propuesto en el código fuente:

```
// Bug actual (index-UvgeZ3yV.js):
a(t.id, e ? `${t.id} - ${e}` : t.id)

// Fix propuesto:
a(e ? `${e}/${t.id}` : t.id, e ? `${t.id} - ${e}` : t.id)
```

Luis Miguel Triana Rueda
Software Engineer · AI Builder
Socorro, Santander, Colombia ■■

linkedin.com/in/luis-triana-2917202a2
github.com/luistriana032006
luistriana.vercel.app