How to Configure Telegram on Clawdbot (Moltbot)

TechRounder PDF Edition

Live article: https://www.techrounder.com/how-to/how-to-configure-telegram-on-clawdbot-moltbot/

By Vipin PG | Published January 28, 2026 | Updated March 9, 2026 | Format: Guide | 6 min read

Quick answer

Create a bot in Telegram via @BotFather to get a bot token, then enable the Telegram channel in your Clawdbot Gateway config (or set TELEGRAM_BOT_TOKEN) and restart. Test by DMing the bot to confirm messages flow, and if you need groups, adjust BotFather privacy/mentions and optionally switch from long-polling to webhook mode.

Key points

This tutorial explains how to set up Telegram in Clawdbot (Moltbot) using the Telegram Bot API, including creating a bot token, configuring Clawdbot via JSON/JSON5 or environment variables, and then verifying that messages are flowing correctly. It notes that Telegram support in Clawdbot is production-ready for bot DMs and groups via grammY, with long-polling enabled by default and webhook mode available as an optional alternative. You'll need a running Clawdbot Gateway environment (server/VPS/local), a Telegram account for testing, and outbound DNS + HTTPS access from the Gateway to 'https://api.telegram.org'. The guide also emphasizes security: your BotFather token is effectively a password, so if it leaks you should immediately revoke/regenerate it in @BotFather and update your configuration.

This tutorial walks you through setting up Telegram on Clawdbot using the Telegram Bot API. You'll learn how to create a Telegram bot token, configure Clawdbot, verify that messages are flowing, and (optionally) enable group features and webhook mode.

Telegram status in Clawdbot: Production-ready for bot DMs + groups (via grammY). Long-polling by default; webhook optional.

Prerequisites

- A running Clawdbot Gateway environment (server, VPS, or local machine where Clawdbot runs).
- Access to the Clawdbot configuration (JSON/JSON5) or environment variables.
- A Telegram account to create and test your bot.
- Outbound network access from your Gateway to 'https://api.telegram.org' (DNS + HTTPS).

Security note: Your Telegram bot token grants control over your bot. Treat it like a password. If it leaks, revoke/regenerate it in BotFather and update your config immediately.

Step 1: Create a Telegram Bot Token (BotFather)

1. Open Telegram and search for @BotFather .
2. Start a chat and run the command: '/newbot'
3. Follow the prompts:
    - Choose a bot display name (e.g., "Clawdbot Assistant").
    - Choose a bot username ending in 'bot' (e.g., 'my_clawdbot_bot').
4. BotFather will return a token that looks like '123456789:ABCDEF...' . Copy it and store it securely.

Optional BotFather Settings (Recommended for Groups)

- ‘/setjoingroups‘ - allow or deny adding the bot to groups.
- ‘/setprivacy‘ - controls whether the bot sees all group messages (Privacy Mode).

Telegram bots default to Privacy Mode ON, meaning they may only receive specific messages in groups (mentions/commands). If you want your bot to respond to all messages in a group, you typically need Privacy Mode OFF or make the bot a group admin.

Step 2: Configure Clawdbot (Token + Channel Settings)

Clawdbot can read the Telegram token either from config or from an environment variable. If both are set, config takes precedence.

Option A (Recommended): Configure Telegram in the Gateway Config

Use a minimal Telegram configuration like this:

```
{
"channels": {
"telegram": {
"enabled": true,
"botToken": "123:abc",
"dmPolicy": "pairing"
}
}
}
```

A slightly more complete example (DM pairing + groups require mention):

```
{
"channels": {
"telegram": {
"enabled": true,
"botToken": "123:abc",
"dmPolicy": "pairing",
"groups": {
"*": { "requireMention": true }
}
}
}
}
```

Option B: Configure Telegram via Environment Variable

Set ‘TELEGRAM_BOT_TOKEN‘ in your environment (works as a fallback for the default Telegram account):

```
export TELEGRAM_BOT_TOKEN="123:abc"
```

After setting the environment variable, start or restart the Gateway so it picks up the new value.

Multi-Account Telegram (Advanced)

If you want multiple Telegram bots, configure per-account tokens using ‘channels.telegram.accounts‘. (The structure matches other multi-account channels in Clawdbot.)

```
{
"channels": {
"telegram": {
"accounts": {
"main": { "botToken": "123:abc" },
"support": { "botToken": "456:def" }
}
}
}
}
```

Step 3: Start/Restart the Gateway

Once the token is configured, start (or restart) the Clawdbot Gateway so the Telegram channel boots. Use whichever command is appropriate for your install:

```
clawdbot gateway restart
```

Telegram starts when a token is resolved (config first, env fallback). If you’re unsure whether it started, check your Gateway logs/status.

```
clawdbot gateway status
clawdbot logs --follow
```

Step 4: DM Access (Pairing vs Allowlist vs Open)

Default (Recommended): Pairing

By default, Clawdbot uses pairing for Telegram DMs: unknown senders receive a pairing code and are ignored until approved. Pairing codes expire after 1 hour.

Example config:

```
{
"channels": {
"telegram": {
"dmPolicy": "pairing"
}
}
}
```

Approve a Pairing Request

1. DM your bot from your Telegram account.
2. Clawdbot will generate a pairing code.
3. On the server running Clawdbot, approve the code:

```
clawdbot pairing list telegram
```

clawdbot pairing approve telegram <CODE>

Allowlist (More Locked Down)

You can restrict DMs to specific user IDs (recommended) or usernames.

```
{
"channels": {
"telegram": {
"dmPolicy": "allowlist",
"allowFrom": ["123456789"]
}
}
}
```

Open (Not Recommended for Most Deployments)

Open access allows any Telegram user who can reach your bot to message it. Use with caution.

```
{
"channels": {
"telegram": {
"dmPolicy": "open",
"allowFrom": ["*"]
}
}
}
```

How to Find Your Telegram User ID

- Safer method: Start the Gateway, DM your bot, then inspect logs for ‘from.id‘ .
- Official Bot API method: DM your bot and fetch updates: ‘curl "https://api.telegram.org/bot<bot_token>/getUpdates"‘ Look for ‘message.from.id‘.
- Third-party bots (less private): ‘@userinfobot‘ or ‘@getidsbot‘ .

Step 5: Using Telegram Groups (Mentions, Allowlists, and Privacy Mode)

Understand Group Controls in Clawdbot

Telegram group behavior in Clawdbot is controlled by two main concepts:

1. Which groups are allowed via ‘channels.telegram.groups‘
    - If ‘groups‘ is not set , all groups are allowed (subject to other policy settings).
    - If ‘groups‘ is set, it becomes an allowlist: only listed groups (or wildcard ‘"*"‘) are accepted.
2. Which senders are allowed via ‘channels.telegram.groupPolicy‘ and ‘channels.telegram.groupAllowFrom‘ Default: ‘groupPolicy‘ is typically ‘"allowlist"‘, meaning groups may appear "silent" until you add allowed senders.
    - ‘groupPolicy: "open"‘ - anyone in allowed groups can talk to the bot.
    - ‘groupPolicy: "allowlist"‘ - only senders in ‘groupAllowFrom‘ can talk to the bot.
    - ‘groupPolicy: "disabled"‘ - no group messages are accepted.

Recommended Group Setup (Mention-Only, Allowlisted Senders)

```
{
"channels": {
"telegram": {
"groupPolicy": "allowlist",
"groupAllowFrom": ["123456789"],
"groups": {
"*": { "requireMention": true }
}
}
}
}
```

Always Respond in a Specific Group

To respond to all messages in a particular group, set ‘requireMention‘ to ‘false‘ for that group ID:

```
{
"channels": {
"telegram": {
"groups": {
"-1001234567890": { "requireMention": false }
}
}
}
}
```

Important: Telegram Privacy Mode

If you expect your bot to read non-mention messages in groups, Telegram’s Privacy Mode must allow it. You usually need to either:

- Disable Privacy Mode via BotFather (‘/setprivacy‘ -> Disable), and remove + re-add the bot to the group; or
- Make the bot a group admin (admin bots receive all messages).

How to Get a Group Chat ID

Telegram group IDs are typically negative numbers like ‘-1001234567890‘. You can obtain the group chat ID by forwarding a message from the group to ‘@userinfobot‘ or ‘@getidsbot‘.

Privacy note: those are third-party bots. If you prefer, inspect your Clawdbot logs after the bot receives a message, or use the official Bot API ‘getUpdates‘ to find ‘chat.id‘.

Quick Activation Toggle (Session-Level)

In a Telegram group, you can toggle activation behavior for that session:

- ‘/activation always‘ - respond to all messages
- ‘/activation mention‘ - require mentions (default)

Note: This is session-level behavior only. For persistence across restarts, set it in config.

Optional: Webhook Mode (Public URL Setup)

By default, Clawdbot uses long-polling (no public URL required). If you prefer webhooks, configure a public webhook URL.

Basic Webhook Configuration

```
{
"channels": {
"telegram": {
"enabled": true,
"botToken": "123:abc",
"webhookUrl": "https://your-domain.example/telegram-webhook"
}
}
}
```

By default, the local listener binds to ‘0.0.0.0:8787‘ and serves ‘POST /telegram-webhook‘. If your public URL differs, place a reverse proxy in front of Clawdbot and route the public path to the local listener.

Webhook Security (Recommended)

You can add a webhook secret and/or customize the webhook path:

```
{
"channels": {
"telegram": {
"webhookUrl": "https://your-domain.example/telegram-webhook",
"webhookSecret": "a-long-random-secret",
"webhookPath": "/telegram-webhook"
}
}
}
```

Verification Checklist

1. Gateway starts cleanly with no Telegram startup errors.
2. DM test: Send a message to the bot in Telegram.
3. Pairing: If ‘dmPolicy‘ is ‘pairing‘ , approve the pairing code and re-test.
4. Group test (optional): Add the bot to a group and mention it (e.g., ‘@YourBot hello‘).
5. Always-respond group test (optional): If you configured ‘requireMention:false‘ , ensure Privacy Mode is OFF or bot is admin.

Troubleshooting

1) Bot menu commands fail / "setMyCommands failed"

This usually indicates network/DNS issues reaching ‘api.telegram.org‘ from your Gateway.

2) Bot responds to mentions but not to normal group messages

- Ensure the group config sets ‘requireMention:false‘ (for that group or wildcard ‘"*"‘).
- Disable Telegram Privacy Mode via BotFather (‘/setprivacy‘), then remove + re-add the bot to the group.
- Or make the bot a group admin.

3) Bot isn’t seeing group messages at all

- If ‘channels.telegram.groups‘ is set, ensure the group ID is listed or wildcard ‘"*"‘ is present.
- Check ‘channels.telegram.groupPolicy‘ and add allowed senders to ‘groupAllowFrom‘ if policy is ‘"allowlist"‘ .
- Verify bot membership and permissions in the group.
- Inspect logs for messages like "skipping group message".

4) Commands like /status don’t work

- Confirm your Telegram user ID is authorized (pairing approval or allowlist).
- Remember: command execution is still authorization-gated, even in "open" groups.

5) Network errors / intermittent failures (IPv6/DNS)

Some environments resolve ‘api.telegram.org‘ to IPv6 first. If your host lacks IPv6 egress, requests can fail or hang. Fix by enabling IPv6 egress or preferring IPv4, then restart the Gateway.

Common Configuration Reference

Minimal DM Pairing Setup

```
{
"channels": {
"telegram": {
"enabled": true,
"botToken": "123:abc",
"dmPolicy": "pairing"
}
}
}
```

Group Defaults (Mention-Only Everywhere)

```
{
"channels": {
"telegram": {
"groups": {
"*": { "requireMention": true }
}
}
}
}
```

Always Respond in All Groups (Use With Caution)

```
{
```

```
"channels": {
"telegram": {
"groups": {
"*": { "requireMention": false }
}
}
}
}
```

Sender Controls (Groups)

```
{
"channels": {
"telegram": {
"groupPolicy": "allowlist",
"groupAllowFrom": ["123456789"]
}
}
}
```

Webhook Toggle

```
{
"channels": {
"telegram": {
"webhookUrl": "https://your-domain.example/telegram-webhook"
}
}
}
```

For the full Telegram channel configuration surface (chunking, media limits, replies, inline buttons, etc.), refer to the Clawdbot Gateway configuration reference and the Telegram channel documentation in your installation.

Next steps: Once Telegram is working, consider tightening security (pairing/allowlists), setting group activation rules, and validating your deployment’s network access to Telegram.