Messaging platforms

Matrix

Matrix is an open, decentralized messaging protocol. OpenClaw connects as a Matrix **user** on any homeserver, so you need a Matrix account for the bot. Once it is logged in, you can DM the bot directly or invite it to rooms (Matrix "groups"). Beeper is a valid client option too, but it requires E2EE to be enabled.

Status: supported via plugin (@vector-im/matrix-bot-sdk). Direct messages, rooms, threads, media, reactions, polls (send + poll-start as text), location, and E2EE (with crypto support).

Plugin required

Matrix ships as a plugin and is not bundled with the core install.

Install via CLI (npm registry):

```
openclaw plugins install @openclaw/matrix
```

Local checkout (when running from a git repo):

```
openclaw plugins install ./extensions/matrix
```

If you choose Matrix during configure/onboarding and a git checkout is detected, OpenClaw will offer the local install path automatically.

Details: Plugins

Setup

1. Install the Matrix plugin:

   From npm: `openclaw plugins install @openclaw/matrix`

   From a local checkout: `openclaw plugins install ./extensions/matrix`

2. Create a Matrix account on a homeserver:

   Browse hosting options at https://matrix.org/ecosystem/hosting/

   Or host it yourself.

3. Get an access token for the bot account:

   Use the Matrix login API with `curl` at your home server:

```
curl --request POST \
  --url https://matrix.example.org/_matrix/client/v3/login \
  --header 'Content-Type: application/json' \
  --data '{
  "type": "m.login.password",
  "identifier": {
    "type": "m.id.user",
    "user": "your-user-name"
  },
  "password": "your-password"
}'
```

   Replace `matrix.example.org` with your homeserver URL.

   Or set `channels.matrix.userId` + `channels.matrix.password` : OpenClaw calls the same login endpoint, stores the access token in `~/.openclaw/credentials/matrix/credentials.json` , and reuses it on next start.

4. Configure credentials:

Env: `MATRIX_HOMESERVER` , `MATRIX_ACCESS_TOKEN` (or `MATRIX_USER_ID` + `MATRIX_PASSWORD`)

Or config: `channels.matrix.*`

If both are set, config takes precedence.

With access token: user ID is fetched automatically via `/whoami` .

When set, `channels.matrix.userId` should be the full Matrix ID (example: `@bot:example.org`).

5. Restart the gateway (or finish onboarding).
6. Start a DM with the bot or invite it to a room from any Matrix client (Element, Beeper, etc.; see https://matrix.org/ecosystem/clients/). Beeper requires E2EE, so set `channels.matrix.encryption: true` and verify the device.

Minimal config (access token, user ID auto-fetched):

```
{
  channels: {
    matrix: {
      enabled: true,
      homeserver: "https://matrix.example.org",
      accessToken: "syt_***",
      dm: { policy: "pairing" },
    },
  },
}
```

E2EE config (end to end encryption enabled):

```
{
  channels: {
    matrix: {
      enabled: true,
      homeserver: "https://matrix.example.org",
      accessToken: "syt_***",
      encryption: true,
      dm: { policy: "pairing" },
    },
  },
}
```

Encryption (E2EE)

End-to-end encryption is **supported** via the Rust crypto SDK.

Enable with `channels.matrix.encryption: true` :

- If the crypto module loads, encrypted rooms are decrypted automatically.
- Outbound media is encrypted when sending to encrypted rooms.
- On first connection, OpenClaw requests device verification from your other sessions.
- Verify the device in another Matrix client (Element, etc.) to enable key sharing.
- If the crypto module cannot be loaded, E2EE is disabled and encrypted rooms will not decrypt; OpenClaw logs a warning.
- If you see missing crypto module errors (for example, `@matrix-org/matrix-sdk-crypto-nodejs-*`), allow build scripts for `@matrix-org/matrix-sdk-crypto-nodejs` and run `pnpm rebuild @matrix-org/matrix-sdk-crypto-nodejs` or fetch the binary with `node node_modules/@matrix-org/matrix-sdk-crypto-nodejs/download-lib.js` .

Crypto state is stored per account + access token in `~/.openclaw/matrix/accounts/<account>/<homeserver>__<user>/<token-hash>/crypto/` (SQLite database). Sync state lives alongside it in `bot-storage.json` . If the access token (device) changes, a new store is created and the bot must be re-verified for encrypted rooms.

Device verification: When E2EE is enabled, the bot will request verification from your other sessions on startup. Open Element (or another client) and approve the verification request to establish trust. Once verified, the bot can decrypt messages in encrypted rooms.

Multi-account

Multi-account support: use `channels.matrix.accounts` with per-account credentials and optional `name` . See `gateway/configuration` for the shared pattern.

Each account runs as a separate Matrix user on any homeserver. Per-account config inherits from the top-level `channels.matrix` settings and can override any option (DM policy, groups, encryption, etc.).

```
{
  channels: {
    matrix: {
      enabled: true,
      dm: { policy: "pairing" },
      accounts: {
        assistant: {
          name: "Main assistant",
          homeserver: "https://matrix.example.org",
          accessToken: "syt_assistant_***",
          encryption: true,
        },
        alerts: {
          name: "Alerts bot",
          homeserver: "https://matrix.example.org",
          accessToken: "syt_alerts_***",
          dm: { policy: "allowlist", allowFrom: ["@admin:example.org"] },
        },
      },
    },
  },
}
```

Notes:

- Account startup is serialized to avoid race conditions with concurrent module imports.
- Env variables (`MATRIX_HOMESERVER` , `MATRIX_ACCESS_TOKEN` , etc.) only apply to the **default** account.
- Base channel settings (DM policy, group policy, mention gating, etc.) apply to all accounts unless overridden per account.
- Use `bindings[].match.accountId` to route each account to a different agent.
- Crypto state is stored per account + access token (separate key stores per account).

Routing model

- Replies always go back to Matrix.
- DMs share the agent’s main session; rooms map to group sessions.

Access control (DMs)

- Default: `channels.matrix.dm.policy = "pairing"`. Unknown senders get a pairing code.
- Approve via:
  - `openclaw pairing list matrix`
  - `openclaw pairing approve matrix <CODE>`
- Public DMs: `channels.matrix.dm.policy="open"` plus `channels.matrix.dm.allowFrom=["*"]`.
- `channels.matrix.dm.allowFrom` accepts full Matrix user IDs (example: `@user:server`). The wizard resolves display names to user IDs when directory search finds a single exact match.
- Do not use display names or bare localparts (example: `"Alice"` or `"alice"`). They are ambiguous and are ignored for allowlist matching. Use full `@user:server` IDs.

Rooms (groups)

- Default: `channels.matrix.groupPolicy = "allowlist"` (mention-gated). Use `channels.defaults.groupPolicy` to override the default when unset.
- Allowlist rooms with `channels.matrix.groups` (room IDs or aliases; names are resolved to IDs when directory search finds a single exact match):

```
{
  channels: {
    matrix: {
      groupPolicy: "allowlist",
      groups: {
        "!roomId:example.org": { allow: true },
        "#alias:example.org": { allow: true },
      },
      groupAllowFrom: ["@owner:example.org"],
    },
  },
}
```

`requireMention: false` enables auto-reply in that room.

`groups."*"` can set defaults for mention gating across rooms.

`groupAllowFrom` restricts which senders can trigger the bot in rooms (full Matrix user IDs).

Per-room `users` allowlists can further restrict senders inside a specific room (use full Matrix user IDs).

The configure wizard prompts for room allowlists (room IDs, aliases, or names) and resolves names only on an exact, unique match.

On startup, OpenClaw resolves room/user names in allowlists to IDs and logs the mapping; unresolved entries are ignored for allowlist matching.

Invites are auto-joined by default; control with `channels.matrix.autoJoin` and `channels.matrix.autoJoinAllowlist`.

To allow **no rooms**, set `channels.matrix.groupPolicy: "disabled"` (or keep an empty allowlist).

Legacy key: `channels.matrix.rooms` (same shape as `groups`).

Threads

Reply threading is supported.

`channels.matrix.threadReplies` controls whether replies stay in threads:

- `off` , `inbound` (default), `always`

`channels.matrix.replyToMode` controls reply-to metadata when not replying in a thread:

- `off` (default), `first` , `all`

Capabilities

Feature	Status
Direct messages	✅ Supported
Rooms	✅ Supported
Threads	✅ Supported
Media	✅ Supported
E2EE	✅ Supported (crypto module required)
Reactions	✅ Supported (send/read via tools)
Polls	✅ Send supported; inbound poll starts are converted to text (responses/ends ignored)
Location	✅ Supported (geo URI; altitude ignored)
Native commands	✅ Supported

Troubleshooting

Run this ladder first:

```
openclaw status
openclaw gateway status
openclaw logs --follow
openclaw doctor
openclaw channels status --probe
```

Then confirm DM pairing state if needed:

```
openclaw pairing list matrix
```

Common failures:

- Logged in but room messages ignored: room blocked by `groupPolicy` or room allowlist.
- DMs ignored: sender pending approval when `channels.matrix.dm.policy="pairing"` .
- Encrypted rooms fail: crypto support or encryption settings mismatch.

For triage flow: .

Configuration reference (Matrix)

Full configuration:

Provider options:

- `channels.matrix.enabled` : enable/disable channel startup.
- `channels.matrix.homeserver` : homeserver URL.
- `channels.matrix.userId` : Matrix user ID (optional with access token).
- `channels.matrix.accessToken` : access token.
- `channels.matrix.password` : password for login (token stored).

- `channels.matrix.deviceName` : device display name.
- `channels.matrix.encryption` : enable E2EE (default: false).
- `channels.matrix.initialSyncLimit` : initial sync limit.
- `channels.matrix.threadReplies` : `off | inbound | always` (default: inbound).
- `channels.matrix.textChunkLimit` : outbound text chunk size (chars).
- `channels.matrix.chunkMode` : `length` (default) or `newline` to split on blank lines (paragraph boundaries) before length chunking.
- `channels.matrix.dm.policy` : `pairing | allowlist | open | disabled` (default: pairing).
- `channels.matrix.dm.allowFrom` : DM allowlist (full Matrix user IDs). `open` requires `"*"` . The wizard resolves names to IDs when possible.
- `channels.matrix.groupPolicy` : `allowlist | open | disabled` (default: allowlist).
- `channels.matrix.groupAllowFrom` : allowlisted senders for group messages (full Matrix user IDs).
- `channels.matrix.allowlistOnly` : force allowlist rules for DMs + rooms.
- `channels.matrix.groups` : group allowlist + per-room settings map.
- `channels.matrix.rooms` : legacy group allowlist/config.
- `channels.matrix.replyToMode` : reply-to mode for threads/tags.
- `channels.matrix.mediaMaxMb` : inbound/outbound media cap (MB).
- `channels.matrix.autoJoin` : invite handling (`always | allowlist | off` , default: always).
- `channels.matrix.autoJoinAllowlist` : allowed room IDs/aliases for auto-join.
- `channels.matrix.accounts` : multi-account configuration keyed by account ID (each account inherits top-level settings).

- `channels.matrix.actions` : per-action tool gating (reactions/messages/pins/memberInfo/channelInfo).

‹ LINE Zalo ›