Networking and discovery

Gateway-Owned Pairing

In Gateway-owned pairing, the **Gateway** is the source of truth for which nodes are allowed to join. UIs (macOS app, future clients) are just frontends that approve or reject pending requests.

Important: WS nodes use **device pairing** (role `node`) during `connect`. `node.pair.*` is a separate pairing store and does **not** gate the WS handshake. Only clients that explicitly call `node.pair.*` use this flow.

Concepts

Pending request: a node asked to join; requires approval.

Paired node: approved node with an issued auth token.

Transport: the Gateway WS endpoint forwards requests but does not decide membership. (Legacy TCP bridge support is deprecated/removed.)

How pairing works

1. A node connects to the Gateway WS and requests pairing.
2. The Gateway stores a **pending request** and emits `node.pair.requested`.
3. You approve or reject the request (CLI or UI).

4. On approval, the Gateway issues a **new token** (tokens are rotated on re-pair).
5. The node reconnects using the token and is now “paired”.

Pending requests expire automatically after **5 minutes**.

CLI workflow (headless friendly)

```
openclaw nodes pending
openclaw nodes approve <requestId>
openclaw nodes reject <requestId>
openclaw nodes status
openclaw nodes rename --node <id|name|ip> --name "Living Room iPad"
```

`nodes status` shows paired/connected nodes and their capabilities.

API surface (gateway protocol)

Events:

- `node.pair.requested` — emitted when a new pending request is created.
- `node.pair.resolved` — emitted when a request is approved/rejected/expired.

Methods:

- `node.pair.request` — create or reuse a pending request.
- `node.pair.list` — list pending + paired nodes.
- `node.pair.approve` — approve a pending request (issues token).
- `node.pair.reject` — reject a pending request.
- `node.pair.verify` — verify `{ nodeId, token }`.

Notes:

`node.pair.request` is idempotent per node: repeated calls return the same pending request.

Approval **always** generates a fresh token; no token is ever returned from `node.pair.request`.

Requests may include `silent: true` as a hint for auto-approval flows.

Auto-approval (macOS app)

The macOS app can optionally attempt a **silent approval** when:

the request is marked `silent`, and

the app can verify an SSH connection to the gateway host using the same user.

If silent approval fails, it falls back to the normal “Approve/Reject” prompt.

Storage (local, private)

Pairing state is stored under the Gateway state directory (default `~/.openclaw`):

`~/.openclaw/nodes/paired.json`

`~/.openclaw/nodes/pending.json`

If you override `OPENCLAW_STATE_DIR`, the `nodes/` folder moves with it.

Security notes:

Tokens are secrets; treat `paired.json` as sensitive.

Rotating a token requires re-approval (or deleting the node entry).

Transport behavior

The transport is **stateless**; it does not store membership.

If the Gateway is offline or pairing is disabled, nodes cannot pair.

If the Gateway is in remote mode, pairing still happens against the remote Gateway's store.

‹ Network model

Discovery and Transports ›