Networking and discovery

Discovery and Transports

OpenClaw has two distinct problems that look similar on the surface:

1. **Operator remote control**: the macOS menu bar app controlling a gateway running elsewhere.
2. **Node pairing**: iOS/Android (and future nodes) finding a gateway and pairing securely.

The design goal is to keep all network discovery/advertising in the **Node Gateway** (`openclaw gateway`) and keep clients (mac app, iOS) as consumers.

Terms

Gateway: a single long-running gateway process that owns state (sessions, pairing, node registry) and runs channels. Most setups use one per host; isolated multi-gateway setups are possible.

Gateway WS (control plane): the WebSocket endpoint on `127.0.0.1:18789` by default; can be bound to LAN/tailnet via `gateway.bind` .

Direct WS transport: a LAN/tailnet-facing Gateway WS endpoint (no SSH).

SSH transport (fallback): remote control by forwarding `127.0.0.1:18789` over SSH.

Legacy TCP bridge (deprecated/removed): older node transport (see Bridge protocol); no longer advertised for discovery.

Protocol details:

- Gateway protocol
- Bridge protocol (legacy)

Why we keep both "direct" and SSH

- **Direct WS** is the best UX on the same network and within a tailnet:
  - auto-discovery on LAN via Bonjour
  - pairing tokens + ACLs owned by the gateway
  - no shell access required; protocol surface can stay tight and auditable
- **SSH** remains the universal fallback:
  - works anywhere you have SSH access (even across unrelated networks)
  - survives multicast/mDNS issues
  - requires no new inbound ports besides SSH

Discovery inputs (how clients learn where the gateway is)

1) Bonjour / mDNS (LAN only)

Bonjour is best-effort and does not cross networks. It is only used for "same LAN" convenience.

Target direction:

- The **gateway** advertises its WS endpoint via Bonjour.

Clients browse and show a “pick a gateway” list, then store the chosen endpoint.

Troubleshooting and beacon details: Bonjour.

Service beacon details

- Service types:
  - `_openclaw-gw._tcp` (gateway transport beacon)
- TXT keys (non-secret):
  - `role=gateway`
  - `lanHost=<hostname>.local`
  - `sshPort=22` (or whatever is advertised)
  - `gatewayPort=18789` (Gateway WS + HTTP)
  - `gatewayTls=1` (only when TLS is enabled)
  - `gatewayTlsSha256=<sha256>` (only when TLS is enabled and fingerprint is available)
  - `canvasPort=<port>` (canvas host port; currently the same as `gatewayPort` when the canvas host is enabled)
  - `cliPath=<path>` (optional; absolute path to a runnable `openclaw` entrypoint or binary)
  - `tailnetDns=<magicdns>` (optional hint; auto-detected when Tailscale is available)

Security notes:

- Bonjour/mDNS TXT records are **unauthenticated**. Clients must treat TXT values as UX hints only.
- Routing (host/port) should prefer the **resolved service endpoint** (SRV + A/AAAA) over TXT-provided `lanHost`, `tailnetDns`, or `gatewayPort`.

TLS pinning must never allow an advertised `gatewayTlsSha256` to override a previously stored pin.

iOS/Android nodes should treat discovery-based direct connects as **TLS-only** and require an explicit "trust this fingerprint" confirmation before storing a first-time pin (out-of-band verification).

Disable/override:

- `OPENCLAW_DISABLE_BONJOUR=1` disables advertising.
- `gateway.bind` in `~/.openclaw/openclaw.json` controls the Gateway bind mode.
- `OPENCLAW_SSH_PORT` overrides the SSH port advertised in TXT (defaults to 22).
- `OPENCLAW_TAILNET_DNS` publishes a `tailnetDns` hint (MagicDNS).
- `OPENCLAW_CLI_PATH` overrides the advertised CLI path.

2) Tailnet (cross-network)

For London/Vienna style setups, Bonjour won't help. The recommended "direct" target is:

- Tailscale MagicDNS name (preferred) or a stable tailnet IP.

If the gateway can detect it is running under Tailscale, it publishes `tailnetDns` as an optional hint for clients (including wide-area beacons).

3) Manual / SSH target

When there is no direct route (or direct is disabled), clients can always connect via SSH by forwarding the loopback gateway port.

See **Remote access**.

Transport selection (client policy)

Recommended client behavior:

1. If a paired direct endpoint is configured and reachable, use it.
2. Else, if Bonjour finds a gateway on LAN, offer a one-tap “Use this gateway” choice and save it as the direct endpoint.
3. Else, if a tailnet DNS/IP is configured, try direct.
4. Else, fall back to SSH.

Pairing + auth (direct transport)

The gateway is the source of truth for node/client admission.

- Pairing requests are created/approved/rejected in the gateway (see **Gateway pairing**).
- The gateway enforces:
  - auth (token / keypair)
  - scopes/ACLs (the gateway is not a raw proxy to every method)
  - rate limits

Responsibilities by component

- **Gateway**: advertises discovery beacons, owns pairing decisions, and hosts the WS endpoint.
- **macOS app**: helps you pick a gateway, shows pairing prompts, and uses SSH only as a fallback.
- **iOS/Android nodes**: browse Bonjour as a convenience and connect to the paired Gateway WS.

‹ Gateway-Owned Pairing Bonjour Discovery ›