Protocols and APIs

Bridge Protocol

The Bridge protocol is a **legacy** node transport (TCP JSONL). New node clients should use the unified Gateway WebSocket protocol instead.

If you are building an operator or node client, use the **Gateway protocol**.

Note: Current OpenClaw builds no longer ship the TCP bridge listener; this document is kept for historical reference. Legacy `bridge.*` config keys are no longer part of the config schema.

Why we have both

- **Security boundary**: the bridge exposes a small allowlist instead of the full gateway API surface.
- **Pairing + node identity**: node admission is owned by the gateway and tied to a per-node token.
- **Discovery UX**: nodes can discover gateways via Bonjour on LAN, or connect directly over a tailnet.
- **Loopback WS**: the full WS control plane stays local unless tunneled via SSH.

Transport

- TCP, one JSON object per line (JSONL).

- Optional TLS (when `bridge.tls.enabled` is true).
- Legacy default listener port was `18790` (current builds do not start a TCP bridge).

When TLS is enabled, discovery TXT records include `bridgeTls=1` plus `bridgeTlsSha256` as a non-secret hint. Note that Bonjour/mDNS TXT records are unauthenticated; clients must not treat the advertised fingerprint as an authoritative pin without explicit user intent or other out-of-band verification.

Handshake + pairing

1. Client sends `hello` with node metadata + token (if already paired).
2. If not paired, gateway replies `error` (`NOT_PAIRED` / `UNAUTHORIZED`).
3. Client sends `pair-request` .
4. Gateway waits for approval, then sends `pair-ok` and `hello-ok` .

`hello-ok` returns `serverName` and may include `canvasHostUrl` .

Frames

Client → Gateway:

- `req` / `res` : scoped gateway RPC (chat, sessions, config, health, voicewake, skills.bins)
- `event` : node signals (voice transcript, agent request, chat subscribe, exec lifecycle)

Gateway → Client:

- `invoke` / `invoke-res` : node commands (`canvas.*` , `camera.*` , `screen.record` , `location.get` , `sms.send`)

- `event` : chat updates for subscribed sessions
- `ping` / `pong` : keepalive

Legacy allowlist enforcement lived in `src/gateway/server-bridge.ts` (removed).

Exec lifecycle events

Nodes can emit `exec.finished` or `exec.denied` events to surface system.run activity. These are mapped to system events in the gateway. (Legacy nodes may still emit `exec.started` .)

Payload fields (all optional unless noted):

- `sessionKey` (required): agent session to receive the system event.
- `runId` : unique exec id for grouping.
- `command` : raw or formatted command string.
- `exitCode` , `timedOut` , `success` , `output` : completion details (finished only).
- `reason` : denial reason (denied only).

Tailnet usage

- Bind the bridge to a tailnet IP: `bridge.bind: "tailnet"` in `~/.openclaw/openclaw.json` .
- Clients connect via MagicDNS name or tailnet IP.
- Bonjour does **not** cross networks; use manual host/port or wide-area DNS-SD when needed.

Versioning

Bridge is currently **implicit v1** (no min/max negotiation). Backward-compat is expected; add a bridge protocol version field before any breaking changes.

‹ Gateway Protocol OpenAI Chat Completions ›