Configuration and operations

Trusted proxy auth

> ⚠ **Security-sensitive feature.** This mode delegates authentication entirely to your reverse proxy. Misconfiguration can expose your Gateway to unauthorized access. Read this page carefully before enabling.

When to Use

Use `trusted-proxy` auth mode when:

- You run OpenClaw behind an **identity-aware proxy** (Pomerium, Caddy + OAuth, nginx + oauth2-proxy, Traefik + forward auth)
- Your proxy handles all authentication and passes user identity via headers
- You're in a Kubernetes or container environment where the proxy is the only path to the Gateway
- You're hitting WebSocket `1008 unauthorized` errors because browsers can't pass tokens in WS payloads

When NOT to Use

- If your proxy doesn't authenticate users (just a TLS terminator or load balancer)
- If there's any path to the Gateway that bypasses the proxy (firewall holes, internal network access)

If you're unsure whether your proxy correctly strips/overwrites forwarded headers

If you only need personal single-user access (consider Tailscale Serve + loopback for simpler setup)

How It Works

1. Your reverse proxy authenticates users (OAuth, OIDC, SAML, etc.)
2. Proxy adds a header with the authenticated user identity (e.g., `x-forwarded-user: nick@example.com`)
3. OpenClaw checks that the request came from a **trusted proxy IP** (configured in `gateway.trustedProxies`)
4. OpenClaw extracts the user identity from the configured header
5. If everything checks out, the request is authorized

Configuration

```
{
  gateway: {
    // Must bind to network interface (not loopback)
    bind: "lan",

    // CRITICAL: Only add your proxy's IP(s) here
    trustedProxies: ["10.0.0.1", "172.17.0.1"],

    auth: {
      mode: "trusted-proxy",
      trustedProxy: {
        // Header containing authenticated user identity (required)
        userHeader: "x-forwarded-user",

        // Optional: headers that MUST be present (proxy verification)
        requiredHeaders: ["x-forwarded-proto", "x-forwarded-host"],

        // Optional: restrict to specific users (empty = allow all)
        allowUsers: ["nick@example.com", "admin@company.org"],
      },
    },
  },
}
```

Configuration Reference

Field	Required	Description
`gateway.trustedProxies`	Yes	Array of proxy IP addresses to trust. Requests from other IPs are rejected.
`gateway.auth.mode`	Yes	Must be `"trusted-proxy"`
`gateway.auth.trustedProxy.userHeader`	Yes	Header name containing the authenticated user identity
`gateway.auth.trustedProxy.requiredHeaders`	No	Additional headers that must be present for the request to be trusted

Field	Required	Description
`gateway.auth.trustedProxy.allowUsers`	No	Allowlist of user identities. Empty means allow all authenticated users.

Proxy Setup Examples

Pomerium

Pomerium passes identity in `x-pomerium-claim-email` (or other claim headers) and a JWT in `x-pomerium-jwt-assertion` .

```
{
  gateway: {
    bind: "lan",
    trustedProxies: ["10.0.0.1"], // Pomerium's IP
    auth: {
      mode: "trusted-proxy",
      trustedProxy: {
        userHeader: "x-pomerium-claim-email",
        requiredHeaders: ["x-pomerium-jwt-assertion"],
      },
    },
  },
}
```

Pomerium config snippet:

```
routes:
  - from: https://openclaw.example.com
    to: http://openclaw-gateway:18789
    policy:
      - allow:
          or:
            - email:
                is: nick@example.com
    pass_identity_headers: true
```

Caddy with OAuth

Caddy with the `caddy-security` plugin can authenticate users and pass identity headers.

```
{
  gateway: {
    bind: "lan",
    trustedProxies: ["127.0.0.1"], // Caddy's IP (if on same host)
    auth: {
      mode: "trusted-proxy",
      trustedProxy: {
        userHeader: "x-forwarded-user",
      },
    },
  },
}
```

Caddyfile snippet:

```
openclaw.example.com {
    authenticate with oauth2_provider
    authorize with policy1

    reverse_proxy openclaw:18789 {
        header_up X-Forwarded-User {http.auth.user.email}
    }
}
```

nginx + oauth2-proxy

oauth2-proxy authenticates users and passes identity in `x-auth-request-email`.

```
{
  gateway: {
    bind: "lan",
    trustedProxies: ["10.0.0.1"], // nginx/oauth2-proxy IP
    auth: {
      mode: "trusted-proxy",
      trustedProxy: {
        userHeader: "x-auth-request-email",
      },
    },
  },
}
```

nginx config snippet:

```
location / {
    auth_request /oauth2/auth;
    auth_request_set $user $upstream_http_x_auth_request_email;

    proxy_pass http://openclaw:18789;
    proxy_set_header X-Auth-Request-Email $user;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
}
```

Traefik with Forward Auth

```
{
  gateway: {
    bind: "lan",
    trustedProxies: ["172.17.0.1"], // Traefik container IP
    auth: {
      mode: "trusted-proxy",
      trustedProxy: {
        userHeader: "x-forwarded-user",
      },
    },
  },
}
```

Security Checklist

Before enabling trusted-proxy auth, verify:

- □ **Proxy is the only path**: The Gateway port is firewalled from everything except your proxy
- □ **trustedProxies is minimal**: Only your actual proxy IPs, not entire subnets

- □ **Proxy strips headers**: Your proxy overwrites (not appends) `x-forwarded-*` headers from clients
- □ **TLS termination**: Your proxy handles TLS; users connect via HTTPS
- □ **allowUsers is set** (recommended): Restrict to known users rather than allowing anyone authenticated

Security Audit

`openclaw security audit` will flag trusted-proxy auth with a **critical** severity finding. This is intentional — it's a reminder that you're delegating security to your proxy setup.

The audit checks for:

- Missing `trustedProxies` configuration
- Missing `userHeader` configuration
- Empty `allowUsers` (allows any authenticated user)

Troubleshooting

"trusted_proxy_untrusted_source"

The request didn't come from an IP in `gateway.trustedProxies` . Check:

- Is the proxy IP correct? (Docker container IPs can change)
- Is there a load balancer in front of your proxy?
- Use `docker inspect` or `kubectl get pods -o wide` to find actual IPs

"trusted_proxy_user_missing"

The user header was empty or missing. Check:

- Is your proxy configured to pass identity headers?

- Is the header name correct? (case-insensitive, but spelling matters)
- Is the user actually authenticated at the proxy?

“trusted*proxy_missing_header**”

A required header wasn’t present. Check:

- Your proxy configuration for those specific headers
- Whether headers are being stripped somewhere in the chain

”trusted_proxy_user_not_allowed”

The user is authenticated but not in `allowUsers` . Either add them or remove the allowlist.

WebSocket Still Failing

Make sure your proxy:

- Supports WebSocket upgrades (`Upgrade: websocket` , `Connection: upgrade`)
- Passes the identity headers on WebSocket upgrade requests (not just HTTP)
- Doesn’t have a separate auth path for WebSocket connections

Migration from Token Auth

If you’re moving from token auth to trusted-proxy:

1. Configure your proxy to authenticate users and pass headers
2. Test the proxy setup independently (curl with headers)
3. Update OpenClaw config with trusted-proxy auth
4. Restart the Gateway

5. Test WebSocket connections from the Control UI
6. Run `openclaw security audit` and review findings

Related

- **Security** — full security guide
- **Configuration** — config reference
- **Remote Access** — other remote access patterns
- **Tailscale** — simpler alternative for tailnet-only access

‹ Authentication Health Checks ›