



☰ Extensions > Voice Call Plugin

Extensions

Voice Call Plugin

Voice calls for OpenClaw via a plugin. Supports outbound notifications and multi-turn conversations with inbound policies.

Current providers:

`twilio` (Programmable Voice + Media Streams)

`telnyx` (Call Control v2)

`plivo` (Voice API + XML transfer + GetInput speech)

`mock` (dev/no network)

Quick mental model:

Install plugin

Restart Gateway

Configure under `plugins.entries.voice-call.config`

Use `openclaw voicecall ...` or the `voice_call` tool

Where it runs (local vs remote)

The Voice Call plugin runs **inside the Gateway process**.

If you use a remote Gateway, install/configure the plugin on the **machine running the Gateway**, then restart the Gateway to load it.

Install



Option A: install from npm (recommended)

>

```
openclaw plugins install @openclaw/voice-call
```

Restart the Gateway afterwards.

Option B: install from a local folder (dev, no copying)

```
openclaw plugins install ./extensions/voice-call  
cd ./extensions/voice-call && pnpm install
```

Restart the Gateway afterwards.

Config

Set config under `plugins.entries.voice-call.config` :



```
plugins: {  
  entries: {  
    "voice-call": {  
      enabled: true,  
      config: {  
        provider: "twilio", // or "telnyx" | "plivo" | "mock"  
        fromNumber: "+15550001234",  
        toNumber: "+15550005678",  
  
        twilio: {  
          accountSid: "ACxxxxxxxx",  
          authToken: "...",  
        },  
  
        telnyx: {  
          apiKey: "...",  
          connectionId: "...",  
          // Telnyx webhook public key from the Telnyx Mission Control Portal  
          // (Base64 string; can also be set via TELNYX_PUBLIC_KEY).  
          publicKey: "...",  
        },  
  
        plivo: {  
          authId: "MAxxxxxxxxxxxxxxxxxxxx",  
          authToken: "...",  
        },  
  
        // Webhook server  
        serve: {  
          port: 3334,  
          path: "/voice/webhook",  
        },  
  
        // Webhook security (recommended for tunnels/proxies)  
        webhookSecurity: {  
          allowedHosts: ["voice.example.com"],  
          trustedProxyIPs: ["100.64.0.1"],  
        },  
      },  
    },  
  },  
}
```



```
// Public exposure (pick one)
// publicUrl: "https://example.ngrok.app/voice/webhook",
// tunnel: { provider: "ngrok" },
// tailscale: { mode: "funnel", path: "/voice/webhook" }

outbound: {
  defaultMode: "notify", // notify | conversation
},

streaming: {
  enabled: true,
  streamPath: "/voice/stream",
},
},
},
},
}
```

Notes:

Twilio/Telnyx require a **publicly reachable** webhook URL.

Plivo requires a **publicly reachable** webhook URL.

`mock` is a local dev provider (no network calls).

Telnyx requires `telnyx.publicKey` (or `TELNYX_PUBLIC_KEY`) unless `skipSignatureVerification` is true.

`skipSignatureVerification` is for local testing only.

If you use ngrok free tier, set `publicUrl` to the exact ngrok URL; signature verification is always enforced.

`tunnel.allowNgrokFreeTierLoopbackBypass: true` allows Twilio webhooks with invalid signatures **only** when `tunnel.provider="ngrok"` and `serve.bind` is loopback (ngrok local agent). Use for local dev only.

Ngrok free tier URLs can change or add interstitial behavior; if `publicUrl` drifts, Twilio signatures will fail. For production,



prefer a stable domain or Tailscale funnel.

Stale call reaper

Use `staleCallReaperSeconds` to end calls that never receive a terminal webhook (for example, notify-mode calls that never complete). The default is `0` (disabled).

Recommended ranges:

Production: `120 - 300` seconds for notify-style flows.

Keep this value **higher than** `maxDurationSeconds` so normal calls can finish. A good starting point is `maxDurationSeconds + 30-60` seconds.

Example:

```
{
  plugins: {
    entries: {
      "voice-call": {
        config: {
          maxDurationSeconds: 300,
          staleCallReaperSeconds: 360,
        },
      },
    },
  },
}
```

Webhook Security

When a proxy or tunnel sits in front of the Gateway, the plugin reconstructs the public URL for signature verification. These options control which forwarded headers are trusted.

`webhookSecurity.allowedHosts` allowlists hosts from forwarding headers.



`webhookSecurity.trustForwardingHeaders` trusts forwarded headers without an allowlist. >

`webhookSecurity.trustedProxyIPs` only trusts forwarded headers when the request remote IP matches the list.

Example with a stable public host:

```
{
  plugins: {
    entries: {
      "voice-call": {
        config: {
          publicUrl: "https://voice.example.com/voice/webhook",
          webhookSecurity: {
            allowedHosts: ["voice.example.com"],
          },
        },
      },
    },
  },
},
}
```

TTS for calls

Voice Call uses the core `messages.tts` configuration (OpenAI or ElevenLabs) for streaming speech on calls. You can override it under the plugin config with the **same shape** – it deep-merges with `messages.tts` .



```
tts: {  
  provider: "elevenlabs",  
  elevenlabs: {  
    voiceId: "pMsXgVXv3BLzUgSXRplE",  
    modelId: "eleven_multilingual_v2",  
  },  
},  
}
```

Notes:

Edge TTS is ignored for voice calls (telephony audio needs PCM; Edge output is unreliable).

Core TTS is used when Twilio media streaming is enabled; otherwise calls fall back to provider native voices.

More examples

Use core TTS only (no override):

```
{  
  messages: {  
    tts: {  
      provider: "openai",  
      openai: { voice: "alloy" },  
    },  
  },  
}
```

Override to ElevenLabs just for calls (keep core default elsewhere):



```

plugins: {
  entries: {
    "voice-call": {
      config: {
        tts: {
          provider: "elevenlabs",
          elevenlabs: {
            apiKey: "elevenlabs_key",
            voiceId: "pMsXgVXv3BLzUgSXRpLE",
            modelId: "eleven_multilingual_v2",
          },
        },
      },
    },
  },
},
},
},
},
}

```

Override only the OpenAI model for calls (deep-merge example):

```

{
  plugins: {
    entries: {
      "voice-call": {
        config: {
          tts: {
            openai: {
              model: "gpt-4o-mini-tts",
              voice: "marin",
            },
          },
        },
      },
    },
  },
},
},
}

```


Inbound calls



Inbound policy defaults to `disabled`. To enable inbound calls, set:

```
{
  inboundPolicy: "allowlist",
  allowFrom: ["+15550001234"],
  inboundGreeting: "Hello! How can I help?",
}
```

Auto-responses use the agent system. Tune with:

`responseModel`

`responseSystemPrompt`

`responseTimeoutMs`

CLI

```
openclaw voicecall call --to "+15555550123" --message "Hello from Oper
openclaw voicecall continue --call-id <id> --message "Any questions?"
openclaw voicecall speak --call-id <id> --message "One moment"
openclaw voicecall end --call-id <id>
openclaw voicecall status --call-id <id>
openclaw voicecall tail
openclaw voicecall expose --mode funnel
```

Agent tool

Tool name: `voice_call`

Actions:

`initiate_call` (message, to?, mode?)



```
continue_call (callId, message)
```

```
speak_to_user (callId, message)
```

```
end_call (callId)
```

```
get_status (callId)
```

This repo ships a matching skill doc at `skills/voice-call/SKILL.md` .

Gateway RPC

```
voicecall.initiate ( to? , message , mode? )
```

```
voicecall.continue ( callId , message )
```

```
voicecall.speak ( callId , message )
```

```
voicecall.end ( callId )
```

```
voicecall.status ( callId )
```

[◀ Community plugins](#)

[Zalo Personal Plugin ▶](#)

Powered by [mintlify](#)