Set up pma-voice and test proximity, radio and calls

pma-voice uses FiveM's internal Mumble voice system. Proximity voice, a framework radio item and a phone-call interface are related but separate features. Test proximity first; a broken radio script does not by itself show that the game server's voice transport is blocked.

Choose a compatible build

Read the author's release notes and the requirements of your phone, radio and framework integrations. The main branch can contain breaking changes before a stable release. Current author notes distinguish calls from older phone-named configuration and state fields, so do not combine an old phone script with new settings blindly. Pin the selected release or commit for a reproducible test.

Back up the current voice resources and configuration before changing them. If your radio or phone integration stores persistent data, also export its matching database. Schedule a short maintenance window, because restarting voice while people are connected can interrupt conversations and complicate the test.

Run one voice implementation

Install the resource with the required name `pma-voice` and remove conflicting startup entries for other voice systems after preserving their configuration. Disable voice features embedded in other menus if the author requires it. Multiple scripts setting proximity and Mumble audio distances can override each other even when both claim to have started successfully.

This example shows a modest native-audio configuration. Check the options against your installed release and place settings before its startup line:

```
setr voice_useNativeAudio true
setr voice_useSendingRangeOnly true
ensure pma-voice
```

Do not enable several alternative audio modes at once. Start the framework's radio and phone integration only after their documented prerequisites. Do not invent an external Mumble IP or open arbitrary voice ports: the normal integrated setup should be checked against the service's assigned networking first. An external voice-server design needs its own approved configuration.

Use two real clients

1. Have two players join and confirm their operating-system input device and FiveM voice settings are correct. Check push-to-talk and microphone privacy permissions.
2. Stand near each other and speak, then move beyond the selected proximity range. Verify both directions; one-way voice can be a client device or script-state issue.
3. Change proximity mode using the installed key binding. Existing players may retain their personal key settings even after a default changes; inspect FiveM key bindings rather than assuming a new default replaced them.
4. Test a shared radio channel separately, then a phone call separately. Record which test failed and any client or server error.

Keep diagnosis narrow

If proximity works but radio does not, inspect the radio resource's integration, item access and channel permissions. If only phone calls fail, compare that script's API expectations with the installed pma-voice version. Avoid restarting the entire framework as a first reaction. For all-player silence after an update, review the last configuration change and competing voice resources.

Disable temporary verbose logging once evidence is captured. Share the voice release, framework integration versions, failing test and redacted error with support. Do not record private conversations as a diagnostic shortcut; a controlled consented microphone test is enough.

Related guides

- Check QBCore dependencies
- Diagnose the assigned game connection

Further reading

- pma-voice author documentation

- pma-voice release history
- pma-radio author project
- Cfx voice scripting reference

Looking for another game? Browse game server setup and troubleshooting guides.

Revision #2
Created 2026-11-09 09:21:20 UTC by CloudExa Hosting Admin
Updated 2026-11-09 09:21:21 UTC by CloudExa Hosting Admin