

Remote Control from Anywhere: Leveraging Open Source and Web Technologies to Operate Your Rig Remotely

Table of Contents

1. Introduction.....	1
2. Why Operate Remotely? (Breaking Free from the Shack)	2
3. What is Open Source? (FOSS vs. Proprietary)	2
4. Hardware Requirements: Older Rigs vs. Modern Transceivers	3
5. The Software Stack Architecture: From Metal to Browser.....	3
6. Core Tooling & Software Links (The Open Source Toolkit)	5
7. Live Demo	6
8. Protecting the Hardware: TX Safeguards & Remote Safety	7
9. Staying Legal: Canadian Regulatory Realities (ISED).....	7
10. Conclusion: Reclaiming Sovereignty & Operating From Anywhere	8
11. Extra Technical Details & Infrastructure Polish	8
12. Open Floor & Q&A	9

1. Introduction

- **Amateur radio has always been about building, experimenting, tinkering, and understanding how things work under the hood.**
- **The Open Source Zealot's View:**
 - Open source isn't just about avoiding licensing fees; it's about **digital sovereignty**.
 - When we rely on closed, proprietary software, we can become dependent on a vendor's decisions about licensing, support, and product lifecycles.
 - When we embrace FOSS (Free and Open Source Software), we gain the freedom to inspect, modify, and maintain our software.
- **The Goal Tonight:** Showing how this philosophy translates into a real-world remote shack, using an IC-7300, Linux, open source software, and web technologies, and how you can do something similar.

2. Why Operate Remotely? (Breaking Free from the Shack)

- **The Shack Freedom:** Moving beyond the physical confines of the radio room: operating from the back deck, the couch on a lazy afternoon, or the lunch room at the office.
 - **Beating Environmental and Practical Constraints:**
 - Bypassing local HOA restrictions or difficult antenna/feedline runs by placing your station hardware where it is practical to install and operate.
 - Setting up a station at the cottage where the HF noise level may be much lower and antenna restrictions may not be an issue.
 - Separating yourself from shack noise, keeping noisy server fans and computer hum out of your operating environment.
 - **Operating on the Go:** Staying active in nets, checking into Winlink email, or chasing DX while travelling or away from home for work. Remote operation is possible wherever you have a suitable internet connection.
 - **The Security & Control Balance:** Doing this securely, leveraging modern VPNs (WireGuard), mesh networks (Tailscale), and other tools to achieve remote access without directly exposing vulnerable radio-control ports to the public internet.
-

3. What is Open Source? (FOSS vs. Proprietary)

- **The Proprietary Model (The "Black Box"):**
 - Software developed and controlled by a vendor, with source code that is generally not available for users to inspect or modify.
 - You may own a copy or a licence to use it, but you cannot modify it, maintain it, or redistribute it.
 - If the company drops support, changes licensing fees, or goes out of business, continued use may become more difficult.
 - **The Open Source Model (FOSS — Free and Open Source Software):**
 - Software distributed under a licence that allows users to access, inspect, modify, and redistribute source code.
 - Driven by collaborative communities, commercial organizations, or a combination of both—in our case, often hams building tools for hams.
 - **Why it matters in Ham Radio:** It aligns perfectly with the spirit of amateur radio experimentation: reclaiming control over our technology and making our gear do cool stuff!
-

4. Hardware Requirements: Older Rigs vs. Modern Transceivers

- **What Your Radio Needs to Go Remote:**
 - **CAT (Computer Aided Transceiver) / Serial Control:** A method to access and change parameters such as frequency and mode and to control PTT. Interfaces include Icom CI-V, serial connections, and USB-based control. Hamlib provides a common control interface for many supported radios.
 - **Audio Pipeline:** A mechanism to move receive and transmit audio back and forth between the radio and the remote operator.
 - **Spectrum / Waterfall Data:** Not all radios expose spectrum data in the same way. Some provide it through supported control protocols or network interfaces; others may require a separate data source or additional hardware.
 - **Modern Radios (The Plug-and-Play Era):**
 - Radios with built-in USB audio and CAT interfaces make computer integration straightforward. The Icom IC-7300, IC-705, and Yaesu FT-710/FTDX10 are examples of radios with computer-control capabilities, although their interfaces and remote-operation features differ.
 - The original IC-7300 uses USB for CAT control and digital audio; it does not have built-in Ethernet networking. A connected computer can bridge the radio to the network.
 - **Older Radios (The Retrofit Approach):**
 - What do you do when you don't have an all-in-one network stream or USB soundcard? You lean on modular software stacks: using a USB soundcard interface (like a Signalink), a serial-to-USB adapter, and a lightweight shack computer (like a Raspberry Pi or old PC running `rigctl`) to bridge older hardware into the modern network.
 - Remember that `rigctl` handles rig-control commands, not audio streaming or waterfall data by itself. Those functions require suitable additional software or hardware.
-

5. The Software Stack Architecture: From Metal to Browser

- **The Layered Approach:** A robust open source remote station isn't just a single app; it's a modular stack where each layer handles one specific job cleanly, following the Unix philosophy.
 - **Layer 1: The Hardware & OS Foundation:**
 - Linux server or another compatible operating system managing USB audio and serial/CAT connections to the radio.
 - The computer acts as the bridge between the physical transceiver and the remote operating interface.
-

- **Layer 2: The Tunnel & Ingress Layer:**

- **WireGuard** or a mesh VPN such as **Tailscale** encrypting network traffic between authorized devices.
- Reverse proxies or **Wiredoor** can provide additional routing to selected services when required.
- A VPN and a reverse proxy serve different purposes: a VPN provides private network connectivity, while a reverse proxy routes application traffic. Not every installation needs both.

- **Layer 3: The Backend Daemons:**

- **rigctld** (Hamlib) or **FLRig** handle supported rig-control commands.
- **wfview/wfweb** provides radio control and audio functionality, either through a client application or web browser.
- **MRRC** provides a FastAPI-based remote-control architecture.
- **Potacat** and its **Echocat** remote-control functionality provide another approach to remote operation.
- Some components can run headlessly, while others require a graphical environment.

- **Layer 4: The Frontend User Interface:**

- Desktop applications such as FLRig and wfview.
 - Browser-based interfaces such as wfweb, Rig-Control-Web, Echocat, and others.
-

VA2BBW Radio Remote Control Network Diagram

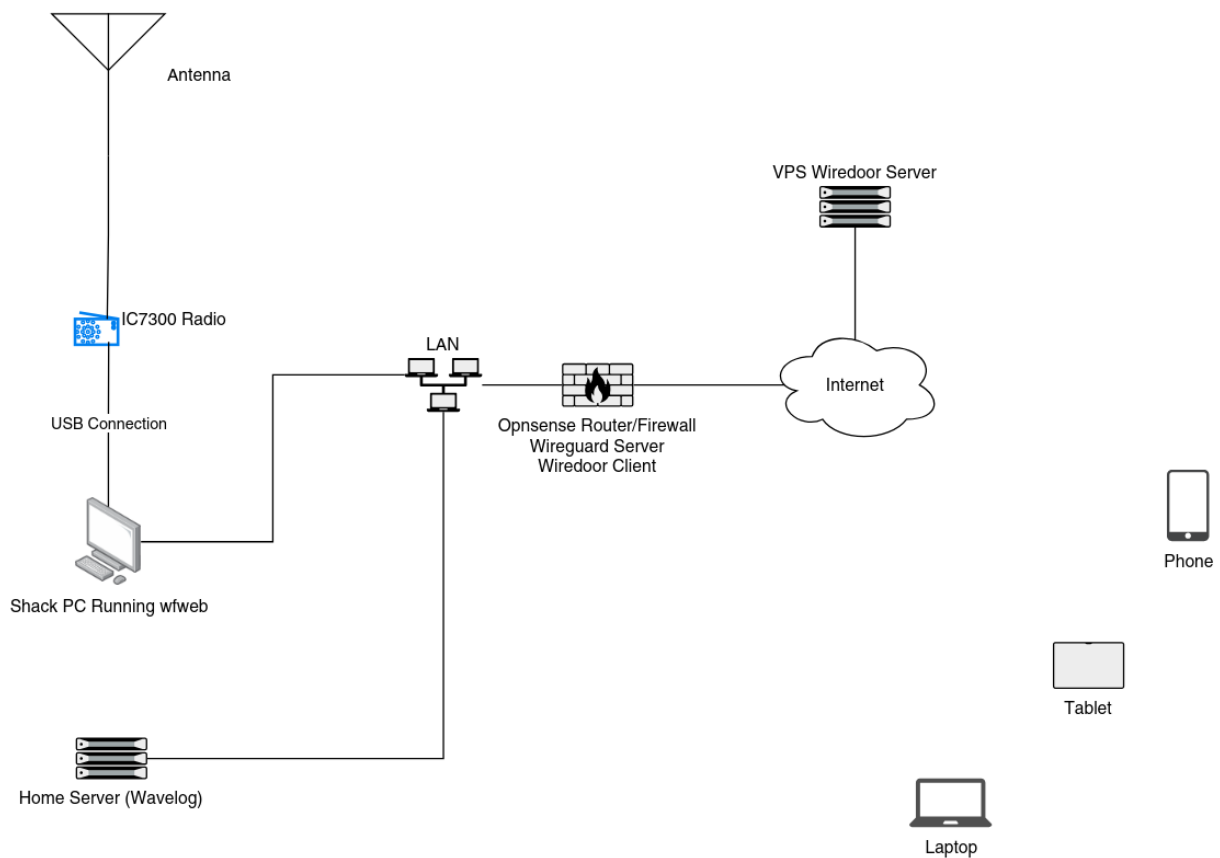


Figure 1. Remote Station Architecture & Data Flow

6. Core Tooling & Software Links (The Open Source Toolkit)

- **Network & Security Tunnels:**
 - **Tailscale:** [Tailscale Website](#) (Managed mesh VPN built around WireGuard for convenient access to authorized devices).
 - **WireGuard:** [WireGuard Project](#) (Fast, modern encrypted VPN tunnelling).
 - **Headscale:** [Headscale Project](#) (Self-hostable coordination server compatible with Tailscale clients).
 - **Wiredoor:** [Wiredoor Project Site](#) (Self-hosted ingress platform using WireGuard tunnels and a reverse proxy to provide access to selected services).
- **CAT Control & Audio Daemons:**
 - **Hamlib / rigctld:** [Hamlib Project Site](#) (Common rig-control library and network-accessible daemon supporting many radios).
 - **Mumble / Murmur:** [Mumble Website](#) (Low-latency voice communication for remote audio).

- **Rig Control & Dashboard Frontends:**

- **wfview:** [wfview Homepage](#) (Open source radio control and audio software for supported Icom and Kenwood radios).
- **wfweb:** [wfweb Repository](#) (Browser-based frontend in the wfview ecosystem for supported Icom radio functions).
- **Rig-Control-Web:** [Rig-Control-Web Project Site](#) (Browser-based rig-control interface with a modular architecture; consult the project documentation for supported backends and radios).
- **Yaesu Web Control (YWC):** [YWC Repository](#) (Browser-oriented CAT and audio control for supported Yaesu equipment).
- **POTACAT / ECHOCAT:** [POTACAT Homepage](#) (POTACAT is an open source POTA/SOTA/DX hunting and activation application; ECHOCAT provides browser/mobile remote-control functionality).
- **MRRC:** [MRRC Homepage](#) (FastAPI-based remote-control system with direct hardware-bus support and documented PTT safeguards).
- **openRig** [openRig](#) (Distro and software for amateur radio control)

- **Miscellaneous:**

- **Pat (Winlink Client):** [Pat Winlink Client](#) (Open source Go-based Winlink client).
- **Wavelog:** [Wavelog Website](#) (Modern, self-hostable web-based amateur radio logbook).

7. Live Demo

- **Walkthrough:** Live demonstration of wfweb running through the configured WireGuard tunnel or Wiredoor infrastructure.
- **Laptop Demo:** Connect to the shack remotely, demonstrate frequency and mode control, and show the waterfall and receive audio.
- **Phone Demo:** Open wfweb on a phone through the WireGuard tunnel and demonstrate the browser-based interface.
- **What to Notice:**
 - The radio remains physically connected to the shack computer.
 - Control commands and audio travel over the network; the radio's RF signal does not.
 - The operator's experience depends on network latency, available bandwidth, and the reliability of the station's internet connection.

8. Protecting the Hardware: TX Safeguards & Remote Safety

- **The Nightmare Scenario:** A frozen browser tab, a dropped cellular connection, or a server-client desynchronization could leave your rig keyed down indefinitely, risking thermal overload, final-stage damage, or worse. An unauthorized user could also gain access to your transmitter.
- **General Hardware Lines of Defense:**
 - **Time-Out Timers (TOT):** Configuring a radio's internal transmit timeout timer, where supported, to force a return to receive after a defined interval.
 - **SWR / Thermal Foldback:** Relying on internal hardware protection mechanisms to reduce power or stop transmission under supported fault conditions. These protections are not substitutes for reliable PTT control.
- **Software-Level Safeguards:**
 - **MRRC Modern's PTT Safety Model:** MRRC Modern documents a layered approach to PTT safety, including server-side safeguards, client heartbeat monitoring, and controls intended to release PTT when the client disconnects. Verify the implementation and behaviour of the version you deploy; no software can guarantee instantaneous recovery from every possible failure.
 - **Application-Level Watchdogs:** Automated cleanup routines in remote-control applications can issue return-to-receive commands upon a socket disconnect or connection timeout. Their effectiveness depends on the implementation and on whether the server and radio remain responsive.
- **Network Security & Access Control:**
 - **Zero Unnecessary Public Exposure:** Never expose raw CAT ports or unauthenticated radio-control interfaces directly to the public internet.
 - **Encrypted Tunnels & Overlays:** Enforce strict access control using private mesh networks like Tailscale, WireGuard tunnels, or self-hosted ingress platforms like Wiredoor. Ensure that only authenticated users and authorized devices can reach the web dashboards and backend sockets.
 - **Test the Failure Modes:** Verify what happens when the browser closes, the remote connection drops, and the control application stops responding. Test PTT release safely before relying on the station remotely.

9. Staying Legal: Canadian Regulatory Realities (ISED)

- **The Couch vs. the Remote Link:** Sitting on your couch with a tablet talking to a radio in the other room via local Wi-Fi may seem harmless, but do not assume that a local network

connection is exempt from ISED's remote-control requirements. The applicable rules do not simply depend on whether the connection uses the internet.

- **The Basic Qualification Restriction:** As detailed in [ISED RIC-3](#), section 4.5, holders of Basic Qualification are not permitted to remotely control fixed stations, regardless of the medium used for control. ISED defines remote control as the ability to indirectly manipulate the technical parameters of a radio, such as bandwidth, emission type, and output power, through an intermediate medium. Operation through a repeater is not considered remote control under this definition.
 - **The Advanced Qualification Privilege:** Remote control of fixed stations is an Advanced Qualification privilege. Basic with Honours provides additional frequency privileges but does not itself grant the Advanced remote-control privilege.
 - **Control Operator Accountability:** Wherever you sit, you remain responsible for compliant transmissions. Understand which privileges apply to your qualification and verify that your proposed remote-control arrangement complies with ISED requirements.
-

10. Conclusion: Reclaiming Sovereignty & Operating From Anywhere

- **Freedom Through Open Source:** By choosing open source tools, you gain the freedom to inspect, adapt, and contribute to the software connecting your radio to the world.
 - **Operating on Your Own Terms:** Whether you're chasing DX from the couch, escaping shack noise, or operating on the go, open source tools and web technologies can make remote operation practical without requiring every component to depend on a vendor's closed ecosystem.
 - **Security Is Part of the Design:** Secure networking, appropriate access controls, and tested PTT safeguards are essential parts of a responsible remote station.
 - **Reviving the Experimenter's Mentality:** Build your own stack, control your own infrastructure, and experiment with tools developed by the amateur radio community.
 - **The Takeaway:** Your radio doesn't have to be where you operate. And your station doesn't have to depend on a vendor's closed tools to make remote operation possible.
-

11. Extra Technical Details & Infrastructure Polish

- **Nginx Reverse Proxies (SSL/TLS etc.):**
 - Running web dashboards behind an Nginx or Caddy reverse proxy allows you to terminate TLS certificates, handle secure WebSockets, and map clean subdomains (e.g., `wfweb.shack.lan`) instead of remembering IP addresses and custom port numbers.
-

- A reverse proxy handles application traffic; it does not automatically provide authentication or secure every connection to the backend.
 - **Dynamic DNS (DDNS) for Residential Connections:**
 - Because many residential fibre/cable connections use dynamic public IP addresses, services such as Cloudflare DDNS, DuckDNS, or a custom script can keep your hostname pointed at your current public IP.
 - DDNS provides a stable name, not security. Remote access still requires appropriate network and application protections.
 - **Local DNS Overrides (LAN Domain Mapping):**
 - Using your router or firewall's DNS service (such as OPNsense Host Overrides, Pi-hole, or AdGuard Home) to point hostnames directly to your server's local IP address.
 - Local DNS overrides can simplify access from inside your LAN and can complement a VPN or reverse-proxy configuration.
-

12. Open Floor & Q&A

- Questions from the floor
-

Get your Advanced. Take control. Go remote!