

RADIO MAGIC

Robert C. Mazur, VA3ROM | www.va3rom.com | va3rom@gmail.com



First published in the July-August 2026 issue of the Canadian Amateur

Take a Wild Ride with UberSDR!

Introduction

To start my 75th TCA column article, I wish outgoing editor Alan Griffin all the best in his “retirement.” I also want to thank Nathan Higgins, M9PSY, the creator of UberSDR, for proofreading this article and adding extra details. I also have to say that I am a total Linux noob (newbie). To me, Linux is not very user-friendly or plug-and-play. A trivial Windows task can often become very convoluted in Linux, such as creating multiple VACs (virtual audio cables) and connecting multiple instances of DSP (digital signal processing) software to them.

I avoided it for decades until one day I happened to be skimming the March 2026 digital issue of Practical Wireless, and Mike Richards, G4WNC’s, article on UberSDR caught my eye because it featured a big picture of the RX-888 MKII SDR (software-defined radio). It has become one of the most popular SDRs for use in many professional radio research projects (**Figure 1**). Unfortunately, the article and software were Linux-based and used the well-known and powerful ka9q-radio open-source software, which is also well-known for being difficult for noobs.

To quote Mike, “Although ka9q-radio is an excellent, highly versatile SDR system, it is essentially a Linux-based command-line application, which limits its adoption to those comfortable working with Linux at that level.”

Meaning that it has no GUI (graphical user interface). I almost stopped there and moved on to the next article, but he continued, “However, thanks to some generously donated work by Nathan, M9PSY, we now have an easy-to-use GUI-based application that utilises the ka9q-radio versatile SDR system.”



Figure 1 — RX-888 MKII SDR with Turn Island Systems GPS clock kit. Credit: rx-888.com and Turn Island Systems.

The article included a link to the UberSDR website (<https://ubersdr.org>), where you will find a simple set of easy-to-follow installation steps. All you have to do is run one bash file — Linux’s MS-DOS batch file equivalent — to automatically install and configure ka9q-radio and the Linux UberSDR server software. No need for dozens of cryptic terminal commands. One button click and done!

I ran into some issues installing the Ubuntu Server version of Linux and used MS Copilot to help me resolve them. But hosting my system on the UberSDR cloud server was a one-click operation, and it is very secure against external attacks.

Was ist “UberSDR”?

The German word “über” (or ueber) generally means “over,” “above,” “beyond,” or sometimes “superior.” But here, it means “super.” With it, you can create a shortwave radio observatory platform without any previous experience or knowledge. You can do casual or serious shortwave radio listening using a web browser (**Figure 2**), monitor Amateur Radio bands, or, in my case, conduct some serious radio science (**Figure 3**). UberSDR is a ride-sharing — I mean radio-sharing — platform built around:

- An RX-888 MKII wideband SDR.
- The ka9q-radio DSP engine.
- A Linux server (Ubuntu or Debian).
- A Windows or Linux web browser (Edge, Chrome, etc.), or the dedicated UberSDR or UberSDR Audio client programs.

UberSDR provides real-time access to the radio spectrum from about 10 kHz to 30 MHz — depending on your antenna system — and can support hundreds of simultaneous users, depending on the “horsepower” and memory of your host Linux server. The intent of UberSDR is to:

- Make wideband shortwave radio monitoring easily accessible.
- Provide built-in digital-mode skimming for FT, JS, CW, WSPR, etc., and real-time web streaming for PSKReporter, WSPRnet, the RBN, etc.
- Create a long-term propagation archive for later statistical analysis.
- Provide live space-weather information through a single integrated dashboard.
- Make it fully open source.
- Make it extremely easy for any Linux noob to install and set up.

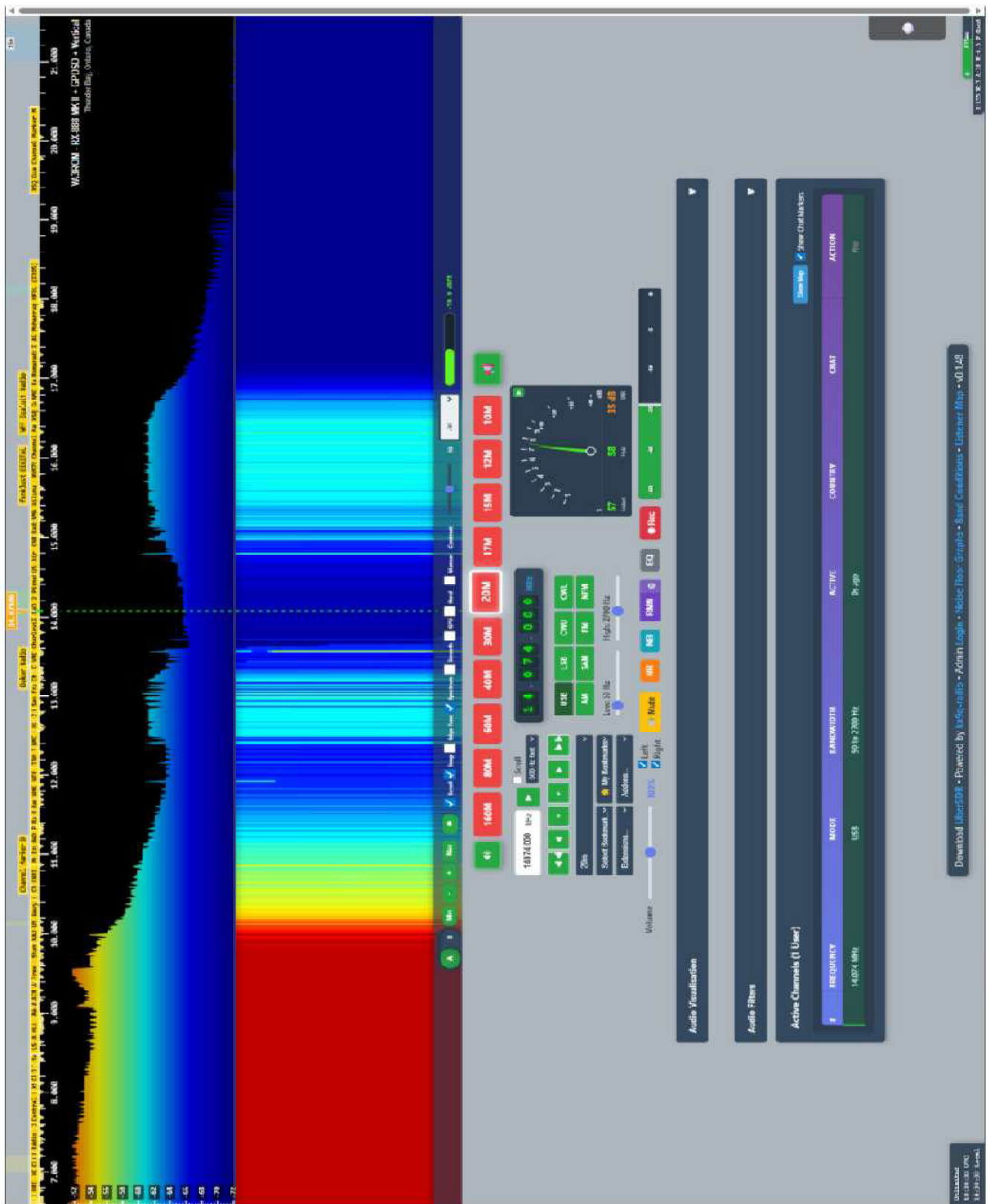


Figure 2 — UberSDR web browser control display. The entire radio spectrum up to 30 MHz is visible. This was mid-morning, so the upper bands were still very “cold” (blue).

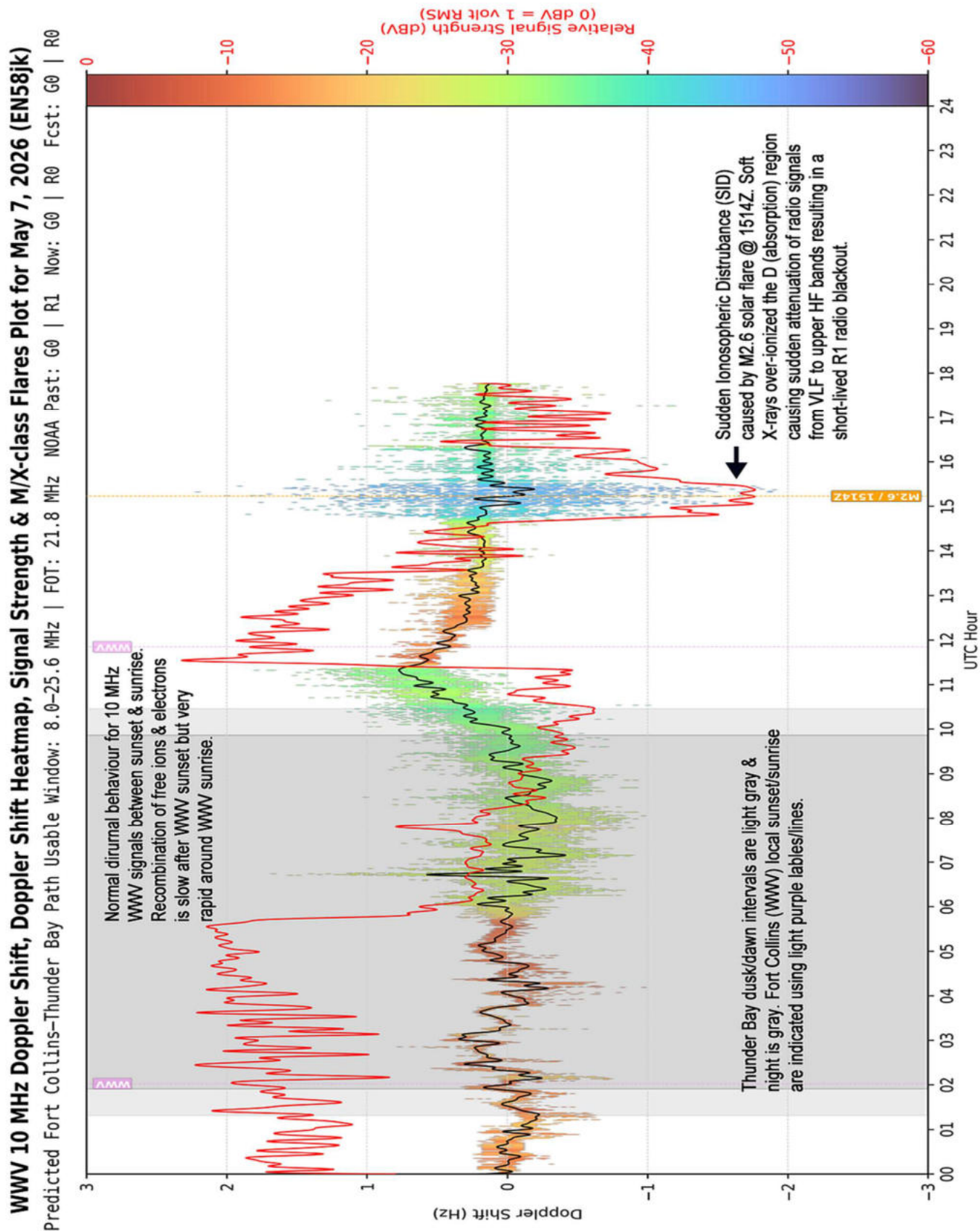


Figure 3 — WWV propagation. One example of what you can do with the UberSDR Audio client and my custom Python scripts. I monitor WWV 24/7 on 5, 10 and 15 MHz.

Driving UberSDR

I will not go into all the technical details because the second part of Mike Richards' article in *Practical Wireless*, March 2026, covers those and explains the entire installation process in minute detail. Suffice it to say, using these two articles and the excellent instructions on the UberSDR website, I did something I never thought possible. UberSDR has so many ancillary features for creating and maintaining statistics that any data “junky” will be kept very busy and very happy (**Figure 4**).

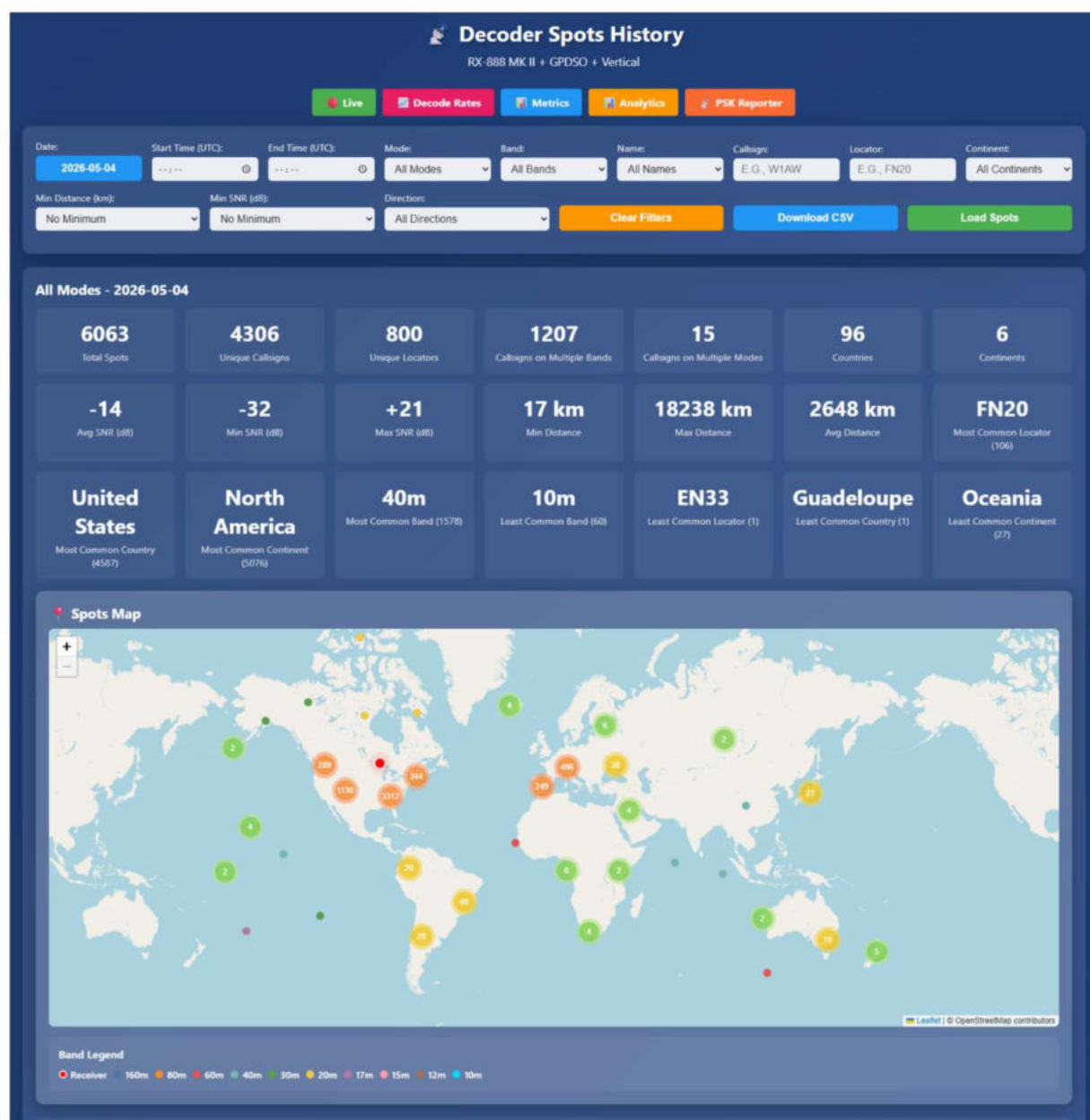


Figure 4 — UberSDR FT8 spots display. A great way to see how well various antennas perform, using FT8 radio signals received across the bands, with a very detailed analysis.

UbersDR runs “headless”, meaning no mouse, keyboard or monitor is connected to the host Linux computer. It runs quietly on my black-box Beelink AMD Ryzen 7, sitting in a corner of my basement radio shack. You j“set it and forget it” (**Figure 5**).



Figure 5 — My UberSDR system.

All my SDRs (top right) are fed by an active 5-port antenna splitter. One RX-888 MKII connects to my Beelink cube computer running Linux, ka9q-radio, and UberSDR (bottom). The other connects to a Windows laptop. In the lower left are three Leo Bodnar GPSDOs and an active 4-port GPS distribution hub. The aluminum stands feature large, USB-powered cooling fans. Electronics last much longer if you let them run 24/7 and keep them cool.

The Linux UberSDR server requires very little, if any, maintenance (unlike Windows), and most maintenance can be performed via a local browser or from anywhere in the world if you host your RX-888 on the UberSDR cloud network. It is very secure, and Nathan has no secret “backdoor” access to anyone’s host servers. You must invite him.

You can use either the Linux or Windows UberSDR client (Figure 6) or the UberSDR Audio client (Figure 7), depending on your requirements. I use the Audio client because it is fast and lightweight, especially when I need multiple instances and do not need the extra features of the full UberSDR client. Both support multiple VAC, IQ signal extraction, and standard radio modes. They also support various DSP ancillary programs, such as WSJT-X and fldigi.

Overall

UberSDR combines high-channel capacity, flexible DSP processing and mode decoding, and user-friendly access, setting it apart from other web-based SDRs. It suits those who want more robust monitoring, extensive database and archival features, and a community-driven approach to radio communications that lets you share your data with others worldwide.

Additionally, UberSDR has built-in port 8073 compatibility for the KiwiSDR, so any KiwiSDR-compatible client program — e.g., KiwiKonnnect (Windows/Mac) or kiwirecorder (Linux) — can connect. UberSDR will then act like a “real” KiwiSDR or Web-888 SDR (also Kiwi-compliant). Perhaps, down the road, Nathan will expand UberSDR to support other SDRs: RSP, Airspy, RTL-SDR, etc.

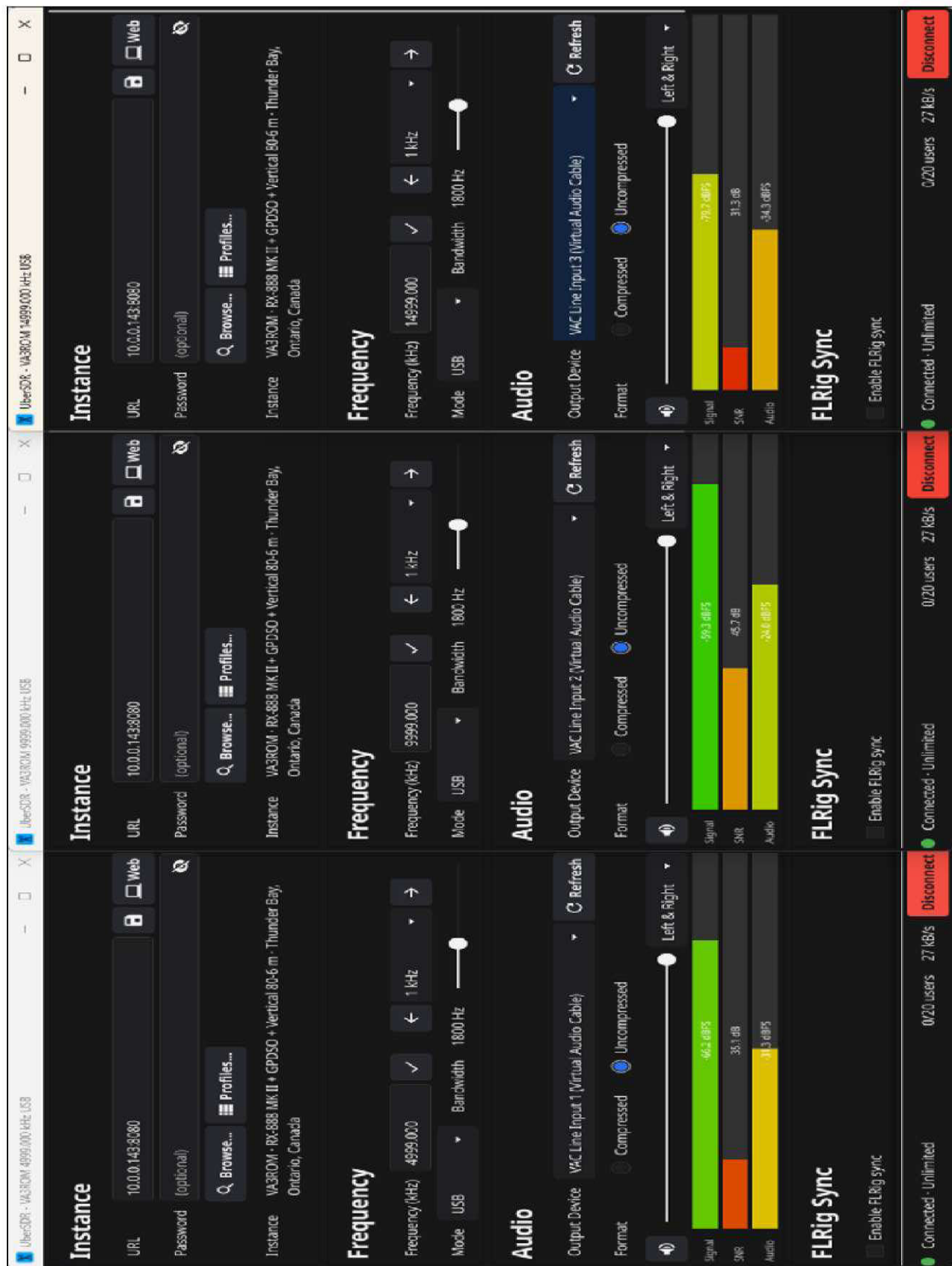


Figure 7 — When you just need the basics, want speed and multiple instances, VACs, DSP software, then use the UberSDR Audio (exe) client.

My Final

I wrote this article to spread the word about UberSDR to Canadian Amateurs and radio hobbyists who may not have heard about it. I also refer you to the two excellent *Practical Wireless* articles that started me on this Linux road trip.

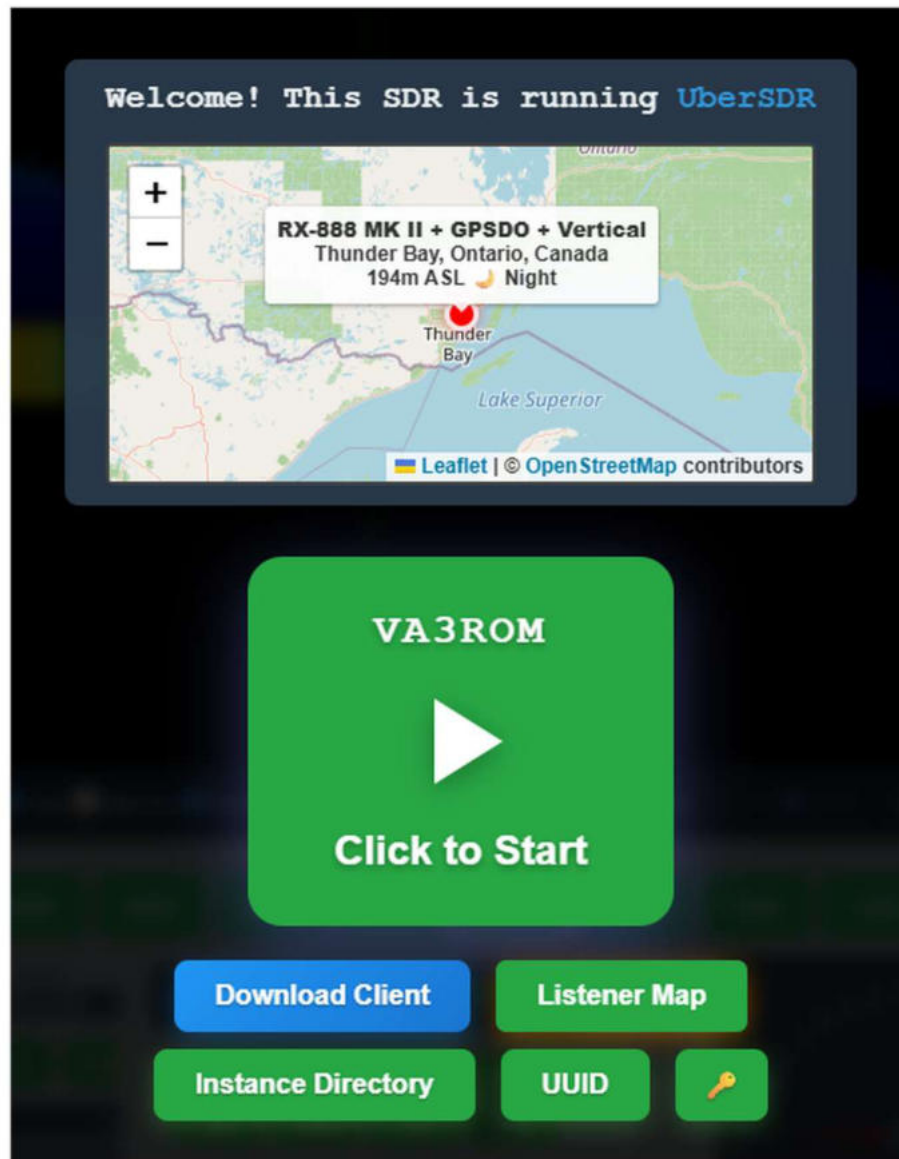


Figure 8 — Welcome to my UberSDR portal.

UberSDR is still in its infancy, but more and more users are hosting new UberSDR systems each month. Many more may not be, because you can also keep your system to yourself. But why not share with the world? It is literally only one button click away (Figure 8). — 73