(alternative) USB-Serial Server

Jim Oberhofer KN6PE
February 2021

Introduction

With my packet station is in the garage and Outpost work going on upstairs in our den, its inconvenient when I need a real RF connection to test Outpost. Having upstairs access to my downstairs TNC and radio would solve that problem. So, when I saw the July 2020 QST Article titled "*A Raspberry Pi USB Serial Server*", I thought this could be the solution!

Mark Erbaugh N8ME, the author of the QST article, does a nice job describing how to tie together a collection of hardware and software to create a USB Serial server (see the QST article appended to this note). However, there is one sentence in his article that turned out to be a problem for me to get it up and running:

> ***"It is distributed as source code only, so it will have to be compiled."***

The "***It***" that Mark refers to is a Windows software program called *com2tcp*. A peek inside one of the downloaded zip files showed the source code was written in Visual C++ and the readme file said it needs to be compiled with Microsoft Visual Studio 2005. Microsoft does offer a free version of Visual C++ to students and individuals (Visual Studio 2019). Unfortunately, I don't have it, never used it, and have too many other projects in the queue to set aside and figure out a new compiler and IDE. The question is: could I find a faster alternative to the code-compiling route?

Alternative

Fortunately, there is. First, a look at what is similar and different between these two approaches:

What's needed	***Original***	***Alternative***	***Implementation notes***
TNC with a serial-to-USB adaptor (or KPC3-Plus USB)	✓	✓	Same for both
Linux server on your LAN	✓	✓	Same for both
A PC on your LAN	✓	✓	Same for both
ser2net*:* A Linux software package that provides a way for a user to connect from a network connection to a serial port.	✓	✓	Same for both
com0com & ***com2tcp****:* Windows programs that creates a virtual PC com port that is mapped to a remote device over a network connection. Source: http://com0com.sourceforge.net/	✓		Requires MS Visual C++ to compile the program '*com2tcp*'
COM2TCP*:* Similar spelling but a different Windows program; see the next paragraph. Source: www.astrogeeks.com		✓	39USD license fee, 45-day free trial

The www.astrogeeks.com website says, "COM2TCP and TCP2COM are applications that allow you to tunnel a serial connection over a network. Although they were designed for remote telescope operation, they can be used to remotely control virtually any serial device." The Windows program we need is **COM2TCP**.

DISCLAIMER: I am not affiliated with Astrogeeks in any way. They just happen to have a solution that I found and worked for me, out of the box.

While I encourage anyone who has Visual C++ experience and a compiler to give the com2tcp compile a try (definitely read the QST article at the end of this note for the details), this *Application Note* implements a USB Serial Server leveraging the solution Mark described for the Linux side and the Astrogeeks *COM2TCP* program on the PC side.

Step 1: Before starting

This application note assumes the following:

1. You have a Linux server with an open USB port. I am using a Raspberry Pi.
2. Your TNC has a USB interface, or a Serial-USB adaptor.
3. You are familiar enough with Linux to install software packages and enter commands.
4. Your Linux server is on your LAN with internet access and you know its IP address.
5. You have a login to the Linux server with admin ('sudo') access.
6. You have a Windows PC is on the same LAN as the Linux server, also with internet access.

Step 2: Set up the Linux server

NOTE: DO NOT CONNECT the TNC to the server yet.

1. Log on to your server and get to the command prompt. In my example, it is "`pi@rp01:~ $`"

```
login as: pi
pi@192.168.1.118's password:
Linux rp01 5.4.72-v7+ #1356 SMP Thu Oct 22 13:56:54 BST 2020 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Jan 25 19:13:12 2021 from 192.168.1.138
pi@rp01:~ $
```

2. Find the serial devices on this server with the **`ls`** command (NOTE: your listing may look different):

```
pi@rp01:~ $ ls /dev/tty*
/dev/tty    /dev/tty19  /dev/tty3   /dev/tty40  /dev/tty51  /dev/tty62
/dev/tty0   /dev/tty2   /dev/tty30  /dev/tty41  /dev/tty52  /dev/tty63
/dev/tty1   /dev/tty20  /dev/tty31  /dev/tty42  /dev/tty53  /dev/tty7
/dev/tty10  /dev/tty21  /dev/tty32  /dev/tty43  /dev/tty54  /dev/tty8
/dev/tty11  /dev/tty22  /dev/tty33  /dev/tty44  /dev/tty55  /dev/tty9
/dev/tty12  /dev/tty23  /dev/tty34  /dev/tty45  /dev/tty56  /dev/ttyAMA0
/dev/tty13  /dev/tty24  /dev/tty35  /dev/tty46  /dev/tty57  /dev/ttyprintk
/dev/tty14  /dev/tty25  /dev/tty36  /dev/tty47  /dev/tty58
/dev/tty15  /dev/tty26  /dev/tty37  /dev/tty48  /dev/tty59
/dev/tty16  /dev/tty27  /dev/tty38  /dev/tty49  /dev/tty6
/dev/tty17  /dev/tty28  /dev/tty39  /dev/tty5   /dev/tty60
/dev/tty18  /dev/tty29  /dev/tty4   /dev/tty50  /dev/tty61
pi@rp01:~ $
```

3. Plug in the TNC USB cable to the server. Enter the **`ls`** command again. Note the new entry, **`/dev/ttyUSB0`** in my case.

```
pi@rp01:~ $ ls /dev/tty*
/dev/tty    /dev/tty19  /dev/tty3   /dev/tty40  /dev/tty51  /dev/tty62
/dev/tty0   /dev/tty2   /dev/tty30  /dev/tty41  /dev/tty52  /dev/tty63
/dev/tty1   /dev/tty20  /dev/tty31  /dev/tty42  /dev/tty53  /dev/tty7
/dev/tty10  /dev/tty21  /dev/tty32  /dev/tty43  /dev/tty54  /dev/tty8
/dev/tty11  /dev/tty22  /dev/tty33  /dev/tty44  /dev/tty55  /dev/tty9
/dev/tty12  /dev/tty23  /dev/tty34  /dev/tty45  /dev/tty56  /dev/ttyAMA0
/dev/tty13  /dev/tty24  /dev/tty35  /dev/tty46  /dev/tty57  /dev/ttyprintk
/dev/tty14  /dev/tty25  /dev/tty36  /dev/tty47  /dev/tty58  /dev/ttyUSB0
/dev/tty15  /dev/tty26  /dev/tty37  /dev/tty48  /dev/tty59
/dev/tty16  /dev/tty27  /dev/tty38  /dev/tty49  /dev/tty6
/dev/tty17  /dev/tty28  /dev/tty39  /dev/tty5   /dev/tty60
/dev/tty18  /dev/tty29  /dev/tty4   /dev/tty50  /dev/tty61
pi@rp01:~ $
```

4. Make sure your raspberry pi is up to date:

```
pi@rp01:~ $ sudo apt update
pi@rp01:~ $ sudo apt dist-upgrade
```

5. Install the ser2net package

```
pi@rp01:~ $ sudo apt install ser2net
```

6. Once ser2net is installed, edit its configuration file with **nano**. With the keyboard's up/down buttons, scroll to the bottom of the file and add the line highlighted below:

```
pi@rp01:~ $ sudo nano /etc/ser2net.conf
                :
        (scroll down to the bottom)
                :
BANNER:banner:\r\nser2net port \p device \d [\s] (Debian GNU/Linux)\r\n\r\n

2000:telnet:600:/dev/ttyS0:9600 8DATABITS NONE 1STOPBIT banner
2001:telnet:600:/dev/ttyS1:9600 8DATABITS NONE 1STOPBIT banner
3000:telnet:600:/dev/ttyS0:19200 8DATABITS NONE 1STOPBIT banner
3001:telnet:600:/dev/ttyS1:19200 8DATABITS NONE 1STOPBIT banner
6400:raw:600:/dev/ttyUSB0:9600 8DATABITS NONE 1STOPBIT
```

The parameters for this line mean the following:

6400	TCP port number of this interface. You will need this when configuring the Windows side. This port number is unregistered with the IANA list port numbers and should not cause any conflict.
raw	type of connection; transfers all data as-is between the connection and the device; essentially a pass-through for the port.
600	Timeout, in seconds
/dev/ttyUSB0	The linux tty device name for this interface found earlier. This must be in the form of /dev/<device>
9600 8DATABITES NONE 1STOPBIT	The serial configuration

When done, press **`cntl-O`**, then Enter, then **`cntl-X`** to exit.

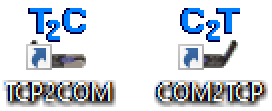
7. Restart ser2net:

```
pi@rp01:~ $ sudo systemctl restart ser2net
pi@rp01:~ $
```

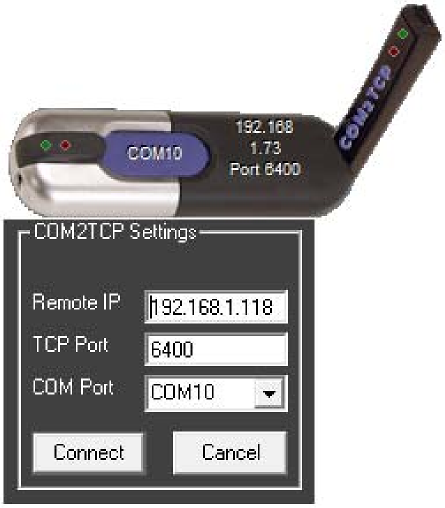
8. The Linux side is set up.

Step 3: Set up the Windows PC

1. From your browser, enter www.astrogeeks.com.
2. Click on **COM2TCP** link on the LEFT.
3. Click on **Download**, then the link for "Download the stable release".
4. Once downloaded, run the installer.
5. Two icons will be placed on your desktop:

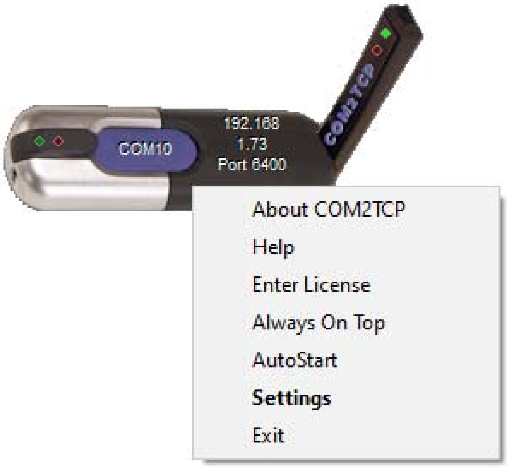


6. Double-click on the C2T COM2TCP icon.
7. Enter the following in the Setting fields:
   a. IP address of your USB serial server.
      Mine was 192.168.1.118.
   b. TCP Port number. This is what you set in the ser2net.conf file above.
   c. Pick an unused COM Port. I am using COM10.
   d. Press Connect.
8. The COM2TCP program will be displayed.



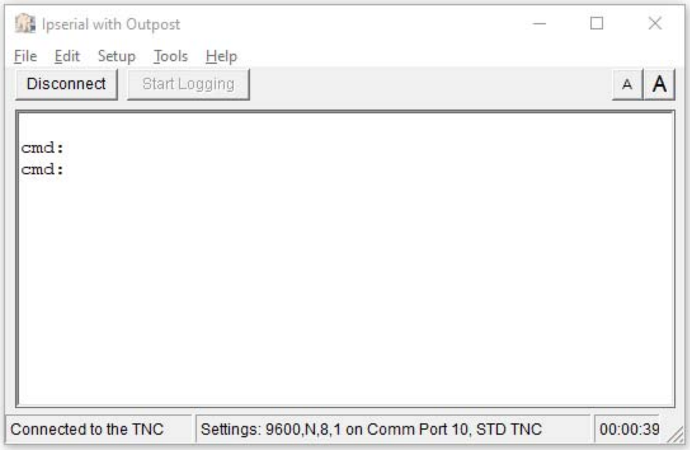
Remote IP 192.168.1.118
TCP Port 6400
COM Port COM10

9. Right click on the C2T COM2TCP device to see various menu options.
   a. **About COM2TCP** will show you the temporary registration key and the date when your trial period is up (***45 days free trial***).
   b. **Help** has a lot more details here on setting C2T up and how it works. This is definitely worth a review.
   c. **AutoStart**. If checked, the program attempts to connect with the previous settings whenever it is started.
   d. Exit. Closes the port, disconnects from the server and closes the program.



Step 4: Test the connection

1. The best first test is to try to connect with a terminal emulator, such as Outpost's Ipserial.exe.
2. Configure Ipserial's com port to match what you set up in COM2TCP, press OK, then Connect.



```
cmd:
cmd:
```

Connected to the TNC | Settings: 9600,N,8,1 on Comm Port 10, STD TNC | 00:00:39

3. If you get the TNC Prompt, it all works! You can now interact with the TNC as if it was directly connected to your local PC.
4. Note the Green and Red virtual 'LEDs' to the left of "COM10". These LEDs seem to indicate the following:
    a. RED: Connect attempted, but the TNC does not appear to be power up, or plugged in.
    b. GREEN: Connect made with the remote device.
    c. OFF/OFF: COM2TCP powered up, not connected.

5. Next Step: try it with Outpost.

Summary

I have had my Raspberry Pi USB Serial Server setup up and running for a couple of weeks now, and other than remembering to turn on the P/S, TNC, and radio in the garage, it has worked flawlessly. I like the idea of network RF packet access from anywhere in the house. With an under-utilized Raspberry Pi sitting around and a 45-day free trial on the Astrogeeks software, this was a no-brainer project for me.

When my 45-day trial runs out, I will spend the 39USD for the Astrogeeks license.