

Givenergy Local Data (GivTCP) Install

This runs through the steps needed to get Docker & GivTCP up and running on a Windows machine although the theory/calls will be similar on all platforms. It first sets up GivTCP in it's most basic form, returning a REST service to prove you can get a response from the inverter. More detailed setup (such as using MQTT etc) are explained later after proof that your system is working has been done.

1: First grab Docker Desktop: [Docker Desktop for Mac and Windows | Docker](#)

2: Install Docker. (You may need to restart.)

3: You may also be prompted to install a secondary package "WSL 2 / Linux kernel update package" - it will direct you to a download page to install it.

4: Hopefully in you should have the Docker software running now  in the notification area.

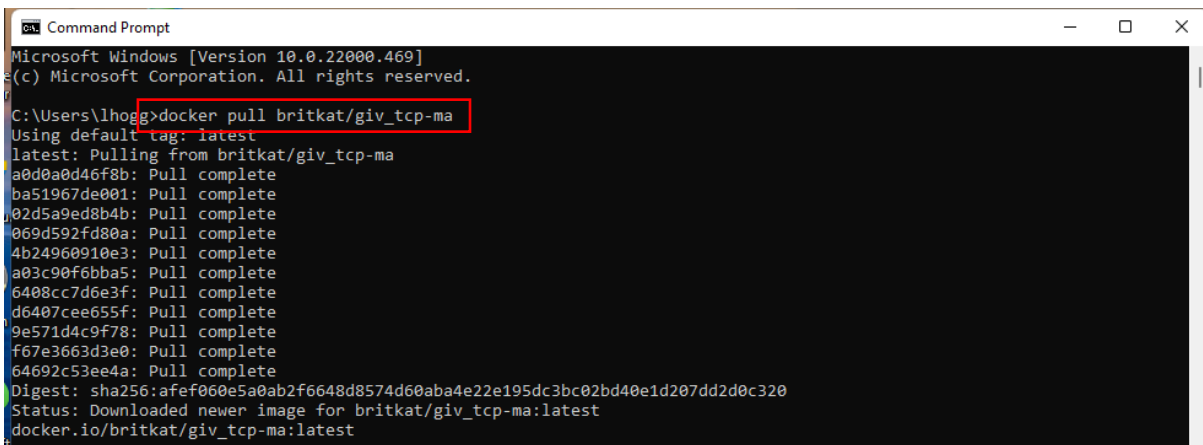
5: Now we need to pull the GivTCP container from Docker Hub which will actually talk to the inverter.

6: Open Command Prompt (Windows Start Menu > search "cmd") which will open the Command Line (black window with white text)

7: copy paste / type in and press enter:

```
docker pull britkat/giv_tcp-ma
```

You should see docker go and fetch the latest version as below



```
Command Prompt
Microsoft Windows [Version 10.0.22000.469]
(c) Microsoft Corporation. All rights reserved.

C:\Users\lhogg>docker pull britkat/giv_tcp-ma
Using default tag: latest
latest: Pulling from britkat/giv_tcp-ma
a0d0a0d46f8b: Pull complete
ba51967de001: Pull complete
02d5a9ed8b4b: Pull complete
069d592fd80a: Pull complete
4b24960910e3: Pull complete
a03c90f6bba5: Pull complete
6408cc7d6e3f: Pull complete
d6407cee655f: Pull complete
9e571d4c9f78: Pull complete
f67e3663d3e0: Pull complete
64692c53ee4a: Pull complete
Digest: sha256:afef060e5a0ab2f6648d8574d60aba4e22e195dc3bc02bd40e1d207dd2d0c320
Status: Downloaded newer image for britkat/giv_tcp-ma:latest
docker.io/britkat/giv_tcp-ma:latest
```

8: Any errors such as "docker not found" usually mean docker isn't installed right. Try to install again & perhaps try a fresh reboot after install.

9: Now we have GivTCP we need to run it (Copy Paste into The Command Line):

(Note: Already know your Inverter IP? - Use the command string in step 11-b to set it directly)

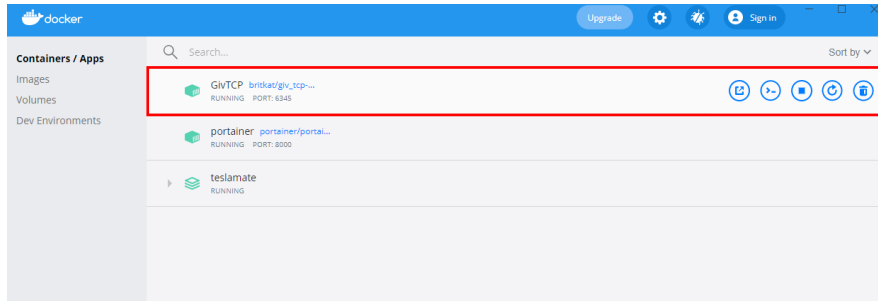
```
docker run --name GivTCP -d -p 6345:6345 britkat/giv_tcp-ma
```

Note: to change the name in Docker change "GivTCP" to whatever you want.

To change the port mapping (if they are already in use) change 6345:6345 to another set of ports.

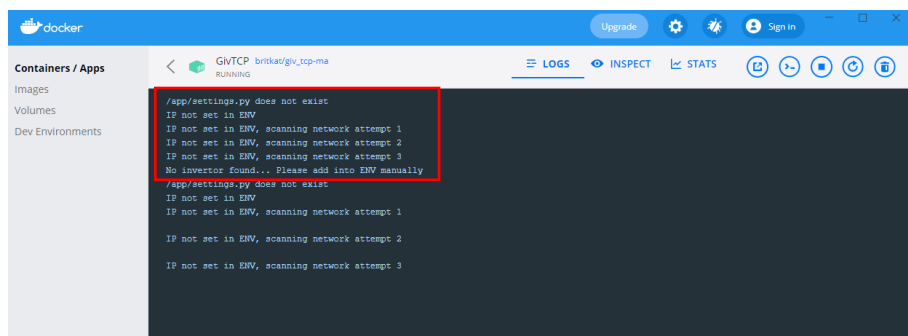
Continued...

10: If you now open Docker, you should see GivTCP running (along with any others you are running) like below:

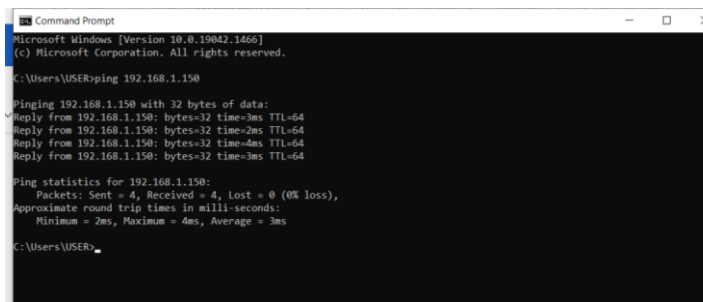


11: If you click on the container, you will see it's command line output. It ***should*** auto discover your inverters IP and start booting up, however this is not always fool proof. If all is good then going to 127.0.0.1:6345/runAll in your browser should bring up data from your inverter.

If it cannot find the inverter on the network (by clicking on the container and getting the error as below) then there are a few options:



- a) If you think you know the IP then you can try to ping the inverter from the cmd window to prove you can connect with it by typing: **ping XXX.XXX.XXX.XXX** or you can try and find it by looking through your routers DHCP list / interface and see if you can see anything with a MAC address starting: **34-ea-e7**



- b) If the above works & you know your Inverters IP address you can delete the container you just created in Docker (Blue recycle bin) & change the Docker run command to point to your known IP and start a new container:

```
docker run --name GivTCP -d -p 6345:6345 -e INVERTOR_IP=XXX.XXX.XXX.XXX britkat/giv_tcp-ma
```

- c) OR if all else fails, you can run my script which will (hopefully) setup everything for you and even work its way through the entire network IP range until it finds the inverter.

https://terravolt.co.uk/Downloads/Misc/Powershell/GivTCP_Auto_Discover_v0.2.bat

Continued...

How the Script works:

1st it returns every IP that is on you machines local ARP table (this is a log of IP's & MAC addresses your machine has seen on the network previously since being switched on). It outputs this as a .txt for you.

2nd it searches through this list looking for a match to the first 3 blocks of the Givenergy WiFi Dongle's MAC address (which in most cases is likely **34-ea-e7**)

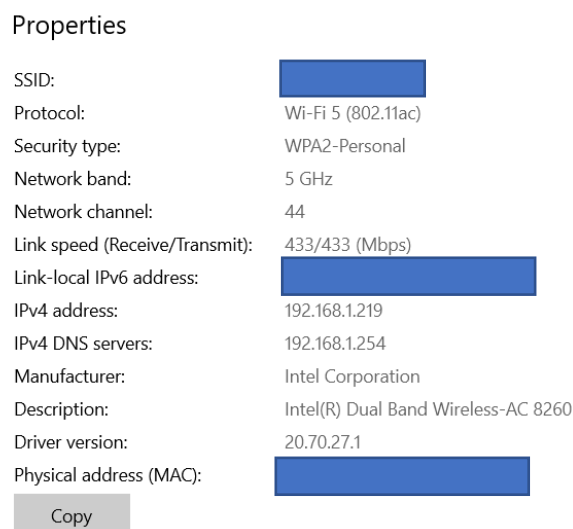
3rd If it finds a match it will offer to download the latest GivTCP from Docker Hub, configure it with the IP address it has found & then start up GivTCP automatically in docker & open the browser to the default data stream.

4th If it can't find the inverter in the current IP list, it will offer to start at the very first IP address on your network and try to ping it. It will increment by the address by 1 each time until it reaches the end of the IP range (1-255). This will populate your machines ARP table with any "live" connected device on your network. It will then start back at step 1 and hopefully then be able match an IP address to your inverter. This will take some time though, so I'd put the kettle on as it works its way through all 255 possible addresses.

Once the script Downloaded, you can simply open it in Notepad and edit the top few lines before running it as follows -

`set dockerName=givtcp_auto` *This is the name you want to call GivTCP in Docker*
`set dockerPort=6345` *This is the port you want to use for GivTCP (default 6345)*
`set range=192.168.1` *This is the first 3 sections of your home networks IP range*

If you don't know your local IP address you can usually find it going to Start > Settings > Network & Internet > "Properties" and then scroll to the bottom.



You should see something along the lines of "IPv4 Address" - in my case this is **192.168.1.219** so I need to use **192.168.1** in my above script to set the range.

Note: no trailing "." is needed

Most standard home routers will probably be either 192.168.0 or 192.168.1 although it could be something completely different depending on the manufacturer.

Calls Available in GivTCP

Further Calls that can be used can be found on the [GivTCP Page](#) but some are shown below. Methods with GET can simply be called up from your browser, Methods with POST will need something like cURL to be used to send a request with a payload.

Read Functions

URL	Method	payload
/runAll	GET	None

Control Functions

URL	Method	payload
/disableChargeTarget	POST	None
/enableChargeTarget	POST	None
/pauseChargeSchedule	POST	None
/resumeChargeSchedule	POST	None
/pauseDischargeSchedule	POST	None
/resumeDischargeSchedule	POST	None
/setChargeTarget	POST	{"chargeToPercent": "50"}
/setBatteryReserve	POST	{"dischargeToPercent": "5"}
/setChargeSlot1	POST	{"start": "0100", "finish": "0400", "chargeToPercent": "55"}
/setChargeSlot2	POST	{"start": "0100", "finish": "0400", "chargeToPercent": "55"}
/setDischargeSlot1	POST	{"start": "0100", "finish": "0400", "dischargeToPercent": "55"}
/setDischargeSlot2	POST	{"start": "0100", "finish": "0400", "dischargeToPercent": "55"}
/setBatteryMode	POST	{"mode": "1"}
/setDateTime	POST	{"dateTime": "dd/mm/yyyy hh:mm:ss"}

Calling the data from another device:

As GivTCP runs as its own webserver, you can call the data up & also make changes from another device on the network by replacing the call to the standard local IP address with the IP address of the device you're running GivTCP / Docker on:

Local:
127.0.0.1:6345/runAll

Becomes (Example Device Running GivTCP) from another device:
192.168.1.150:6345/runAll

Note: You may need to allow Docker through your Firewall if you are not getting a response.

Continued....

Environment Variables & Usage

So we've already touched on how to set an Environment Variable in docker, however this was just for forcing the Inverter IP. Most will also want to use MQTT or send the output to a database etc... So you can set further variables to do that as below:

```
docker run --name GivTCP -d -p 6345:6345 -e INVERTOR_IP=XXX.XXX.XXX.XXX britkat/giv_tcp-ma
```

Further Environment variables can be set by simply adding in as many `-e %Your Environment Variable=Whatever%` as you need to the docker run string you call from the cmd line. Remember you may need to delete your old / running GIVTCP Docker first (or give it a new name)

ENV Name	Example	Description
INVERTOR_IP	192.168.10.1	Docker container can auto detect Invertors if running on your host network. If this fails then add the IP manually to this ENV
NUMBATTERIES	1	Number of battery units connected to the inverter
MQTT_OUTPUT	True	Optional if set to True then MQTT_ADDRESS is required
MQTT_ADDRESS	127.0.0.1	Optional (but required if OUTPUT is set to MQTT)
MQTT_USERNAME	bob	Optional
MQTT_PASSWORD	cat	Optional
MQTT_TOPIC	GivEnergy/Data	Optional - default is Givenergy.
LOG_LEVEL	Error	Optional - you can choose Error, Info or Debug. Output will be sent to the debug file location if specified, otherwise it is sent to stdout
DEBUG_FILE_LOCATION	/usr/pi/data	Optional

ENV Name	Example	Description
PRINT_RAW	False	Optional - If set to True the raw register values will be returned alongside the normal data
INFLUX_OUTPUT	False	Optional - Used to enable publishing of energy and power data to influx
INFLUX_TOKEN	abcdefg123456789	Optional - If using influx this is the token generated from within influxdb itself
INFLUX_BUCKET	giv_bucket	Optional - If using influx this is data bucket to use
INFLUX_ORG	giv_tcp	Optional - If using influx this is the org that the token is assigned to