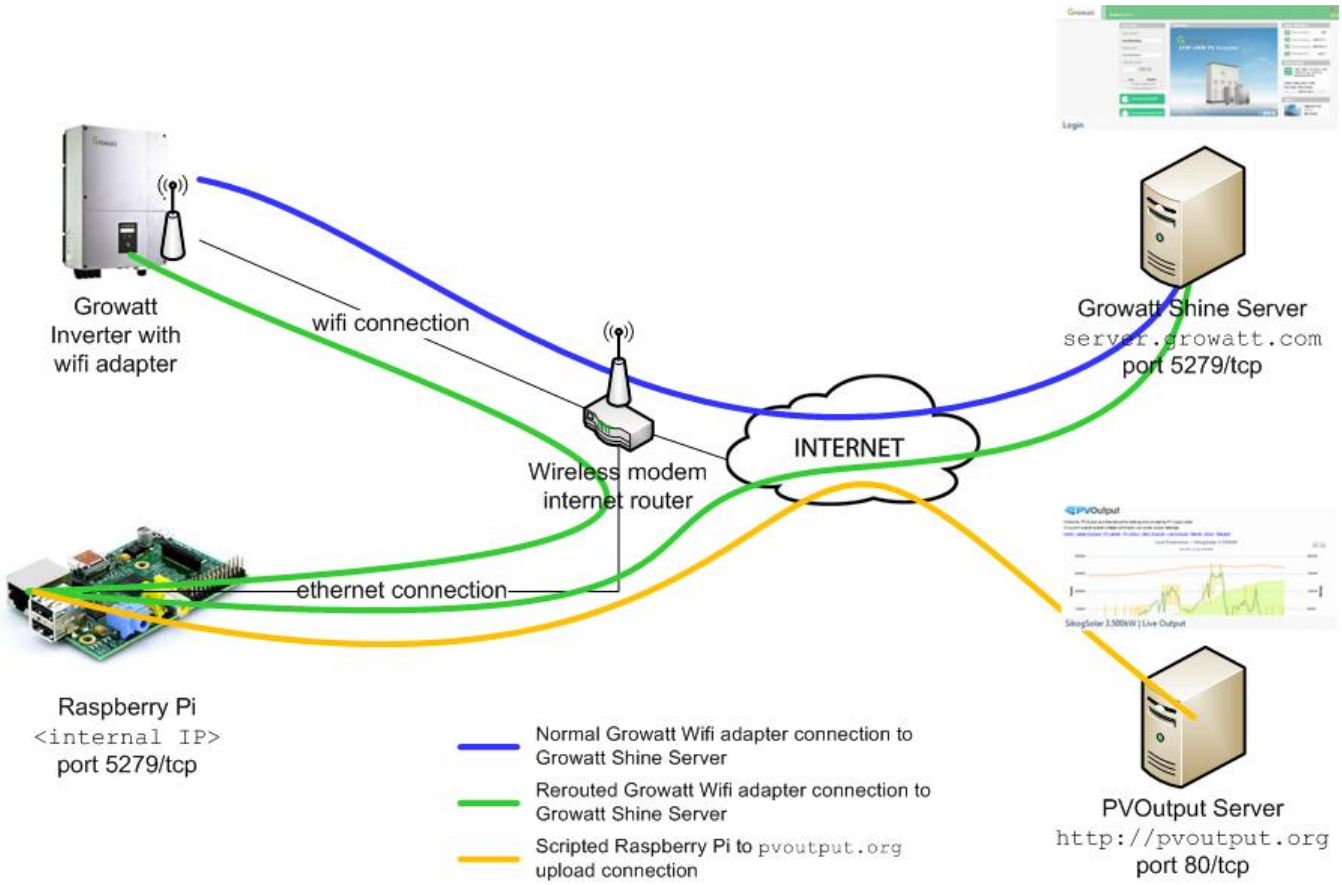
Automatic Growatt Wifi Module data upload to `pvoutput.org` with Raspberry Pi



Author: Sander Plug
energiecommunity / tweakers

Date: 09 July 2015
Version: 2.3

Index

1 Growatt Wifi Module to `pvoutput.org` upload

Finally I have found a workable solution to automatically upload the production data from my Growatt 3600MTL (dual tracker) interver to `http://www.pvoutput.org`. Most solutions documented on the internet were around the Growatt Bluetooth adaptor. Or it involved a USB to RS232 serial cable in combination with a Raspberry Pi closely placed to the Growatt inverter.

Initial struggles to see if the Growatt Wifi Module could be read from the network lead to no results.

Recently I received a solution from jvdmast (tweakers) who manage to capture the data and load this into Pvoutput automatically. This solution does the base necessities and included no error checking and/or documentation. The script needed to be modified to capture the data correctly for my inverter (due to difference in data position in the captured record for my inverter). I updated the solution to include dynamic data analysis (to find the right offset position), error handling and to create a full step-by-step discription which can be used to by others with the same setup. I created a new script to view all the known data from a particular Growatt data record, which can be used for debugging purposes, or future use in storing this information in an own database. I used information from Growatt and JT (energiecommunity) for this. I would like to thank jvdmast and JT for testing the solution and provided valuable feedback/input.

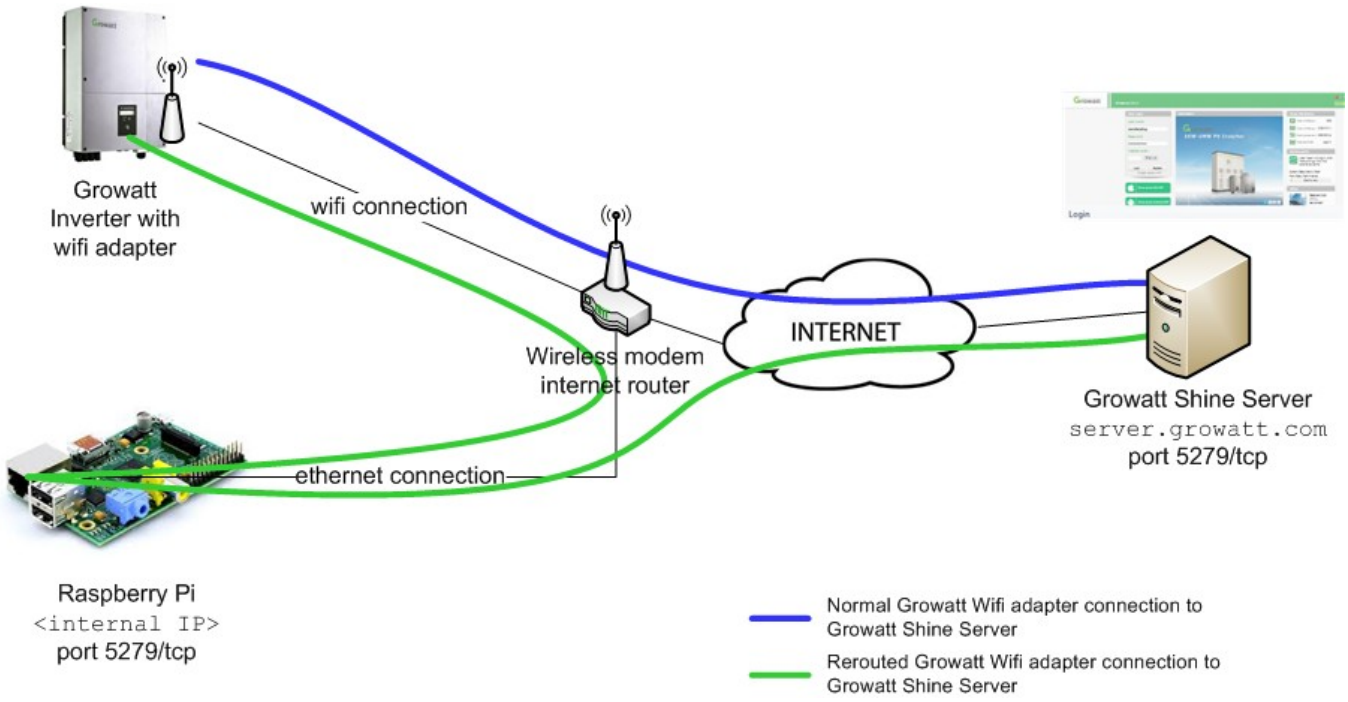
This version (2.3) is the third major release that adds additional feedback received during the last year.
It accomodates for and additional Growatt Wifi Module 3.0.0.0 firmware version. The capture code and iptables rules are now embedded in the `process_growatt_pvoutput.sh` script.
Growatt announced that it will migrate the sever to a new location. The process script does now accomodate for a changing IP address of the `server.growatt.com` server.

It is assumed that the Growatt Wifi Module is currently configured correctly into the local network and that the upload to the `server.growatt.com` server is working correctly.

1.1 Growatt Wifi Module traffic rerouting

The Growatt Wifi Module is typically configured to send the data directly to `server.growatt.com` server. This is configured within the web interface of the Growatt Wifi Module. To be able to capture the traffic and data from the Growatt Wifi Module this traffic needs to be rerouted to the Raspberry Pi.

The following diagram shows how the traffic is rerouted (green flow) via the Raspberry Pi to the `growatt.server.com` server. The normal traffic is shown in blue.



1.2 Port forwarding with iptables on the Raspberry Pi

With iptables it is possible to route the Growatt Wifi Module via the Raspberry Pi to the `server.growatt.com` server. The following commands are needed to forward all traffic to port 5279 on the Raspberry Pi to port 5279 on `server.growatt.com`.

Note: Growatt announced that it will migrate the server.growatt.com sever to a new location. This may change the future IP address from 42.121.252.160 into something else. Be sure to update the rules accordingly.

Run the following commands to enable port forwarding on the Raspberry Pi:

```
sudo echo "1" > /proc/sys/net/ipv4/ip_forward
sudo iptables -t nat -A PREROUTING -p tcp --dport 5279 -j DNAT --to-destination 42.121.252.160:5279
sudo iptables -t nat -A POSTROUTING -j MASQUERADE
```

Run the following command to see if the iptables forwarding rules are active:

```
sudo iptables -t nat -L
```

The output should look like the following.

```
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
DNAT       tcp  --  anywhere             anywhere             tcp dpt:5279 to:42.121.252.160:5279

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
MASQUERADE  all  --  anywhere             anywhere
```

Permanently save these firewall rules so the become active again after a reboot.

```
sudo bash -c 'iptables-save > /etc/network/iptables'
```

Also the the `/etc/network/interfaces` file needs to be edited to activate the saved iptables configuration from `/etc/network/iptables`

```
sudo vi /etc/network/interfaces
```

Add the following line at the end of the file:

```
pre-up iptables-restore < /etc/network/iptables
```

To ensure that the forwarding is enabled after a reboot as well, uncomment the next line to enable packet forwarding for IPv4.

```
sudo vi /etc/sysctl.conf

# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
```

1.3 Configure the Growatt Wifi Module

The Growatt Wifi Module is by default configured to communicate direct to `server.growatt.com`. This needs to be changed to so that the Growatt Wifi Module points to the (internal) IP address of the Raspberry Pi. The forward rules will ensure that from there the traffic is re-routed to the `server.growatt.com` servers as normal.

Note: Growatt has been known to reconfigure the `Server Address(IP Address or Domain Name)` field back to the value of `server.growatt.com`a If the data is not flowing anymore to http://www.pvoutput.org, but it is to http://server.growatt.com, then check if the value is still correctly configured.

Go to the `http://<Growatt Wifi Module IP>/sta_config.html` page and fill out the (internal) IP address of the Raspberry Pi server in the `Server Address(IP Address or Domain Name)` field.

After changing of the address in the Growatt Wifi Module a reboot of the adapter is needed to make it active.

Once the Growatt Wifi adapter is back up again go to `http://<Growatt Wifi Module IP>/sta_config.html` and log back in to the Growatt Wifi Module to check if the connection to `server.growatt.com` is re-established.

Note: The above instructions assumes that the Growatt Wifi Module was already correctly configured to work with an existing account on http://server.growatt.com

1.4 Install Required Programs

Processing of the Growatt Wifi Module data requires that `bc` and `editcap` are installed.

The `bc` binary is used to do calculations, HEX to decimal conversion and is installed by the following command.

```
sudo apt-get install bc
```

The `editcap` binary, part of Wireshark package, is needed to split the capture files into individual files per package to ensure they can be analyzed correctly.

```
sudo apt-get install wireshark
```

The `tcpdump` binary is used to capture the Growatt (data) packages and is installed by the following command.

```
sudo apt-get install tcpdump
```

1.5 Capture Growatt Wifi Module to Growatt Server data

1.5.1 Create Directory Structure

To store the scripts, input- and processed files and log files create the following directory structure.

```
sudo mkdir /home/pvoutput
sudo mkdir /home/pvoutput/logs
sudo mkdir /home/pvoutput/input
sudo mkdir /home/pvoutput/processed
sudo mkdir /home/pvoutput/scripts
sudo mkdir /home/pvoutput/tmp
```

1.6 Install `process_growatt_pvoutput.sh`

Place the `process_growatt_pvoutput.sh` into `/home/pvoutput/scripts` directory.

The following cron entry makes sure that the data capture files are processed every 5 minutes and send to the `pvoutput.org` server.

```
sudo vi /etc/cron.d/growatt_pvoutput
```

Add the following content to the file to run the script every 5 minutes throughout the day:

```
# Run the main processing script every 5 minutes
*/5 * * * * root /home/pvoutput/scripts/process_growatt_pvoutput.sh >/dev/null 2>&1
```

As an alternative, since there is usually no production at night, it is possible to stop processing from 23:00 hours to 05:00 hours. For this use the following content within the cron file (instead of the one above).

```
# Run the main processing script every 5 minutes from 05:00 - 23:00 hours.
*/5 5-22 * * * root /home/pvoutput/scripts/process_growatt_pvoutput.sh >/dev/null 2>&1
```

Congratulations! In principle this is the last step to get the data flowing...

It is however recommended to also perform the steps in the next chapter as this will help keeping the produced output and log files on the Raspberry Pi manageable.

1.7 Cleanup routines

The capturing and processing of the Growatt Wifi Module data generates quite a number of data files and log information. Not all of this information should be kept for a long time. To prevent the filesystem running full the following steps are recommended to perform.

1.7.1 Cleanup data files

To remove the processed, bad and to large data files add the following items to the crontab entry file.

```
sudo vi /etc/cron.d/growatt_pvoutput
```

Add the following content to the file:

```
# Remove - correctly processed files - older then 7 days
30 0 * * * root find /home/pvoutput/processed -name "growatt*ok" -type f -mtime +7 -exec rm {} \;

# Remove - original capture files, which split into new files - older then 30 days
30 0 * * * root find /home/pvoutput/processed -name "growatt*split" -type f -mtime +30 -exec rm {} \;

# Remove - files not correctly uploaded - older then 30 days
30 0 * * * root find /home/pvoutput/processed -name "growatt*badupload" -type f -mtime +30 -exec rm
{} \;

# Remove - original capture files that were to small - older then 30 days
30 0 * * * root find /home/pvoutput/processed -name "growatt*size" -type f -mtime +30 -exec rm {} \;
```

1.7.2 Log file rotation

Rotate the processing log file, compress them and keep a maximum of 7 version.

```
sudo vi /etc/logrotate.d/growatt_pvoutput
```

Add the following content to the file

```
/home/pvoutput/logs/growatt_process.log {
        daily
        rotate 7
        compress
}
```

1.8 Known issues

This chapter describes the known issues and possible local work-arounds.

1.8.1 `h#!/bin/sh` or `#!/bin/bash`

The scripts are written to use `/bin/sh`. Certain Raspberry Pi installations may not have `/bin/sh`. In this case you can change the `#!/bin/sh` directive to `#!/bin/bash`.

1.8.2 Error message: Forbidden 403: Exceeded 60 requests per hour

Especially when setting up the connection and testing that everything works correctly, you may run into the following error message:

```
Forbidden 403: Exceeded 60 requests per hour
```

The `pvoutput.org` servers limited the amount of uploads/requests to 60 requests per hour for normal accounts. The script has been updated to exit when this error message is detected. This error condition should resolve itself.

1.8.3 E_Today counter restarts at 0 when the Growatt inverter is restarted

The Growatt inverter itself does not have an internal clock. Each time the inverter starts it “expects” that it is the start of a new day. Most of the times it will be the start of a new calendar day as well. However restarts of the Growatt inverter may also occur during the day. This could be due to bad weather or user needed to restart the inverter. The result of the restart is that the E_Today counter restarts at 0 (zero) again and the total production in `pvoutput.org` for that day will be incorrect. Restarts of the Growatt inverter during the day would normally not happen that frequently.

From `process_growatt_pvoutput.sh` version 2.1 it is possible to use E_Total to upload to PVOutput. To enable this ensure the `PVCUMFLAG` within the script is set to 1 (`PVCUMFLAG=1`).

2 Migration instructions

2.1 Migrate from version 2.1

These instructions describe the changes to be done to move to migrate to version 2.1

2.1.1 Install new `process_growatt_pvoutput.sh`

Place the new (version 2.3) `process_growatt_pvoutput.sh` into `/home/pvoutput/scripts` directory. This replaces/overwrites the existing (version 2.1) of the `process_growatt_pvoutput.sh` script.

2.1.2 Remove the capture script

Some of the file names and extensions have changed. To ensure these files are cleaned as well the cleanup routine lines need to be updated the following into the crontab entry file.

```
sudo rm /home/pvoutput/scripts/capture_growatt_traffic.sh
```

Remove the `capture_growatt_traffic.sh` script from the `/etc/rc.local` file so it will not be called after a reboot.

```
sudo vi /etc/rc.local
```

Remove the following lines from the `/etc/rc.local` file.

```
# Starts capturing Growatt Wifi Module traffic
/home/pvoutput/scripts/capture_growatt_traffic.sh
```

3 Change Log

3.1 Version 2.3 – released 09-07-2015

Added support for Growatt Wifi Module 3.0.0.0 firmware. Added solution to accomodate for chaning ip-address of the `server.growatt.com` destination. Set cumulative flag (`PVCUMFLAG`) default to 1 (enabled). Added tcpdump logic improvement from Menno Norden to avoid promiscuous mode messages in the `syslog` (-p) and set the recording to 300 seconds with a check if it is still running. This avoids data gaps.

3.2 Version 2.1 – released 22-06-2014

The documentation was updated to include crontab instructions for only running the processing between 5-22 (inclusive) hours.
Also updated instructions are provided for the cleanup routines to match the new file extension.
Support is added for the PVOutput c1 option. With this cumulative flag it is possible to use the E_Total generation number instead of the E_Today number. This can be helpful if the inverter is restarted during the day, caused by insufficient power (e.g. bad weather) or restarts during failures. Normally the E_Today counter starts at zero again, whereas E_Today keeps counting.
Added data position correction within in the `calcpos` function that takes into account the Growatt Wifi Module firmware version.
Removed the fixed path for curl, now it is assumed it can be found in the `$PATH` like the other commands.
Various changes in logfile messages and formatting. Also changed to file names are made during processing to bring clarity. This also excludes capture files from processing if they were not preprocessed first.

3.3 Version 2.0 – released 14-06-2014

Completely reorganized captured packet validation and pre-processing. Larger files are now split into one file per packet. Validation of valid files is more thorough to ensure no data is missed. The previous version and method had discarded valid data in case of communication issues with `server.growatt.com` server. When communication was restarted valid data was mixed with other pakets and not considered for processing. This version splits the capture file into individual files, one per packet, which are all processed.
Added options and code to also upload Vpv1 or Vpv1/Vpv2 data to PVOutput.

3.4 Version 1.0 – released 10-06-2014

Initial version.

4 Growatt Data References

The data of a Growatt inverter is stored within registers. Each register has its own values and meaning assigned to it. These registers are passed within the data record send to the `server.growatt.com` server. The “tcpdump offset from Growatt Serial” column contains the relative offset from the first character of the Growatt inverter serial number found in the tcpdump record. Some records are single HEX words, whereas other records have a high (H) and low (L) HEX word.

This information is found in “Growatt PV Inverter Modbus RS485 RTU Protocol V3.04.pdf” from 2013-02-02.

Register NO.	Variable Name	Description	Value	Unit	Note	tcpdump offset from Growatt Serial
0	Inverter Status	Inverter run state	0:waiting 1:normal 2:fault			15
1	Ppv H	Input power (high)		0.1W		17
2	Ppv L	Input power (low)		0.1W		
3	Vpv1	PV1 voltage		0.1V		21
4	PV1Curr	PV1 input current		0.1A		23
5	PV1Watt H	PV1 input watt (high)		0.1W		25
6	PV1Watt L	PV1 input watt (low)		0.1W		
7	Vpv2	PV2 voltage		0.1V		29
8	PV2Curr	PV2 input current		0.1A		31
9	PV2Watt H	PV2 input watt (high)		0.1W		33
10	PV2Watt L	PV2 input watt (low)		0.1W		
11	Pac H	Output power (high)		0.1W		37
12	Pac L	Output power (low)		0.1W		
13	Fac	Grid frequency		0.01Hz		41
14	Vac1	Three/single phase grid voltage		0.1V		43
15	Iac1	Three/single phase grid output current		0.1A		45
16	Pac1 H	Three/single phase grid output watt (high)		0.1VA		47
17	Pac1 L	Three/single phase grid output watt (low)		0.1VA		
18	Vac2	Three phase grid voltage		0.1V		51
19	Iac2	Three phase grid output current		0.1A		53
20	Pac2 H	Three phase grid output power (high)		0.1VA		55
21	Pac2 L	Three phase grid output power (low)		0.1VA		
22	Vac3	Three phase grid voltage		0.1V		59
23	Iac3	Three phase grid output current		0.1A		61

24	Pac3 H	Three phase grid output power (high)		0.1VA		63
25	Pac3 L	Three phase grid output power (low)		0.1VA		
26	Energy today H	Today generate energy (high)		0.1KWH		67
27	Energy today L	Today generate energy today (low)		0.1KWH		
28	Energy total H	Total generate energy (high)		0.1KWH		71
29	Energy total L	Total generate energy (low)		0.1KWH		
30	Time total H	Work time total (high)		0.5S		75
31	Time total L	Work time total (low)		0.5S		
32	Temperature	Inverter temperature		0.1C		79
33	ISO fault Value	ISO Fault value		0.1V		81
34	GFCI fault Value	GFCI fault Value		1mA		83
35	DCI fault Value	DCI fault Value		0.01A		85
36	Vpv fault Value	PV voltage fault value		0.1V		87
37	Vac fault Value	AC voltage fault value		0.1V		89
38	Fac fault Value	AC frequency fault value		0.01 Hz		91
39	Temperature fault Value	Temperature fault value		0.1C		93
40	Fault code	Inverter fault bit	&*1			95
41	IPM Temperature	The inside IPM in inverter Temperature		0.1C		97
42	P Bus Voltage	P Bus inside Voltage		0.1V		99
43	N Bus Voltage	N Bus inside Voltage		0.1V		101
44	Check Step	Product check step			Reserved	??? – 103
45	IPF	Inverter output PF now	0-20000			??? – 105
46	ResetCHK	Reset check data	1 to reset		Reserved	??? – 107
47	DeratingMode	DeratingMode	0:no deratring; 1:PV; 2:; 3:Vac; 4:Fac; 5:Tboost; 6:Tinv; 7:Control; 8:*LoadSpeed; 9:*OverBackBy Time		“*”is Reserved	??? – 109
48	Epv1_today H	PV Energy today				115
49	Epv1_today L	PV Energy today		0.1kWh		
50	Epv1_total H	PV Energy total				119
51	Epv1_total L	PV Energy total		0.1kWh		
52	Epv2_today H	PV Energy today				123

53	Epv2_today L	PV Energy today		0.1kWh		
54	Epv2_total H	PV Energy total				127
55	Epv2_total L	PV Energy total		0.1kWh		
56	Epv_total H	PV Energy total				131
57	Epv_total L	PV Energy total		0.1kWh		
58	Rac H	AC Reactive power				135
59	Rac L	AC Reactive power		0.1Var		
60	E_rac_today H	AC Reactive energy				139
61	E_rac_today L	AC Reactive energy		0.1kVarh		
62	E_rac_total H	AC Reactive energy				143
63	E_rac_ total L	AC Reactive energy		0.1kVarh		
64	WarningCode	Warning Code				
65	WarningValue	Warning Value				

The following are the inverter fault codes as referenced in the table above by `&*1`.

5 Script sources and information

5.1 Diagnosis script: `view_growatt_data.sh`

This script was created to get (known) data from a capture record. It has been tested on a Growatt 3600MTL, 4400TL and other models. There might be differences in where the data is stored. With this script it can be verified if the data in the captured record corresponds with the data in the `server.growatt.com` server.

```
sudo /home/pvoutput/scripts/view_growatt_data.sh \
 /home/pvoutput/processed/growatt_20140606_18\:22_55.preproc.ok
```

This will give output in the following format.

```
--------------------------------------------------------------------------------
Mon Jun  9 12:33:50 CEST 2014
--------------------------------------------------------------------------------
Growatt Inverter serial    (CS00000000)                Capture sample date : 20140608
Growatt Wifi Module serial (AH00000000)                Capture sample time : 10:53
Growatt Inverter status:    normal (1)                 Growatt temperature   37.5 C
--------------------------------------------------------------------------------
E_Today          1.2 kWh       E_Total       1785.1 kWh       Total time    1868.0 hrs
--------------------------------------------------------------------------------
Ppv            898.5  W        Pac            858.0  W        Fac            50.02 Hz
--------------------------------------------------------------------------------
Vpv1           210.2  V        Vpv2           202.5  V
Ipv1             2.1  A        Ipv2             2.0  A
Ppv1           432.2  W        Ppv2           466.3  W
--------------------------------------------------------------------------------
Vac1           238.2  V        Vac2             0.0  V        Vac3             0.0  V
Iac1             3.5  A        Iac2             0.0  A        Iac2             0.0  A
Pac1           858.0  W        Pac2             0.0  W        Pac2             0.0  W
--------------------------------------------------------------------------------
Epvtotal      1842.4 kW
Epv1today        0.6 kW        Epv2today        0.7 kW
Epv1total      881.8 kW        Epv2total      960.6 kW
--------------------------------------------------------------------------------
ISO Fault        0.0  V        Vpvfault         0.0  V        Tempfault        0.0  C
GFCI Fault       0.0 mA        Vacfault         0.0  V        Faultcode             0
DCI Fault        0.0  A        Facfault        0.00 Hz
--------------------------------------------------------------------------------
IPMtemp          0.0  C        Rac              0.0 Var
Pbusvolt       394.7  V        E_Rac_today      0.0 Var
Nbusvolt         0.0  V        E_Rac_total      0.0 Var
--------------------------------------------------------------------------------
```

The following is the source code of the script.

Ensure that the user variables are changed according to your local setup (e.g. CS00000000 as serial number and 1.0.0.0 / 2.0.0.0 / 3.0.0.0 as Growatt Wifi Module firmware version):

```
GROWATTSERIAL="<your Growatt inverter serial number>"
GROWATTMODULEVER=1.0.0.0
```

```
#!/bin/sh
#------------------------------------------------------------------------
# Name      : view_growatt_data.sh
#
# Function  : Read Growatt Wifi Adapter records and show all contents
#
# Date      : 09-07-2015
# Author    : S. Plug
#
# Version   : 2.3 Added support for Growatt Wifi Module 3.0.0.0 firmware.
#             2.2 Not released
#             2.1 Initial version
#
#------------------------------------------------------------------------

# Uncomment to have debugging information
# set -x

#------------------------------------------------------------------------
# Declaration of user variables
#------------------------------------------------------------------------
GROWATTSERIAL="CS31511376"
GROWATTMODULEVER=1.0.0.0

#------------------------------------------------------------------------
# Declaration of script variables
#------------------------------------------------------------------------
PVBASEDIR=/home/pvoutput
PVINDIR=$PVBASEDIR/input
PVOUTDIR=$PVBASEDIR/processed
PVCAPFILE="growatt_*.cap"
LOWERLIMIT=300
UPPERLIMIT=700
DIVIDER="-----------------------------------"
fn=$1

#------------------------------------------------------------------------
# Check if the file exists
#------------------------------------------------------------------------
if [ ! -f "$fn" ]; then
        printf "\nERROR: The file \"%s\" can not be found...\n\n" $fn
        printf "Usage: %s <Growatt tcpdump file>\n\n" $0
        exit 0
fi

#------------------------------------------------------------------------
# Function to calculate register position based on $OFFSET and parameter.
# There are different Growatt Wifi Module firmware versions.
# The difference between the positions of the position of GROWATTSERIAL
# and the position of the data is +6 positions when 2.0.0.0 firmware or
# 3.0.0.0 firmware is # used (compared to 1.0.0.0 firmware).
#------------------------------------------------------------------------
calcpos()
{
        case "$GROWATTMODULEVER" in
        "1.0.0.0")
                OFFSETCORRECT=0
        ;;
        "2.0.0.0")
                OFFSETCORRECT=6
        ;;
        "3.0.0.0")
                OFFSETCORRECT=6
```

```
        ;;
        esac

        echo $(echo $OFFSET + $1 + $OFFSETCORRECT | bc)
}

#------------------------------------------------------------------------
# Print header
#------------------------------------------------------------------------
printf "%87s\n" | tr ' ' -
date
printf "%87s\n" | tr ' ' -

#------------------------------------------------------------------------
# Determine if the captured record has the desired length
#------------------------------------------------------------------------
fs=$(stat --format=%s $fn)

if [ "$fs" -gt $LOWERLIMIT -a "$fs" -lt $UPPERLIMIT ]; then

        #--------------------------------------------------------------------
        # Use $GROWATTSERIAL to find the reference point to be used as offset
        #--------------------------------------------------------------------
        OFFSET=$(grep -obUaP \
               $(echo -n "$GROWATTSERIAL" | od -A n -t x1 | sed "s/ /\\\x/g") $fn \
                | cut -d ":" -f 1)

        if [ "$OFFSET" = "" ];
        then
                #----------------------------------------------------------------------------
                # Write error message if the file is of an incorrect length
                #----------------------------------------------------------------------------
                printf "\nERROR: Growatt serial number %s not found in datarecord.\n\n" \
                        $GROWATTSERIAL
                exit 0
        fi

        #--------------------------------------------------------------------
        # Fill the variables for Date (fd) and Time (ft)
        #--------------------------------------------------------------------

        fd=$(echo $fn | cut -d_ -f2)
        ft=$(echo $fn | cut -d_ -f3 | cut -c1-5)

        #--------------------------------------------------------------------
        # Extract Growatt Inverter and Adapter serial numbers
        #--------------------------------------------------------------------
        cs=$(hexdump -C -s ${OFFSET}                   -n 10 $fn | cut -d "|" -f 2 | head -1)
        ah=$(hexdump -C -s $(echo ${OFFSET} -10 | bc)  -n 10 $fn | cut -d "|" -f 2 | head -1)

        #--------------------------------------------------------------------
        # Extract Growatt Inverter information
        #--------------------------------------------------------------------

        # Inverter Status (0: waiting, 1: normal, 2:fault)
        InvStatt=$(hexdump -C -s $(calcpos 15) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
        InvStat=$(echo "ibase=16;obase=A;scale=1;$InvStatt" | bc)

                case $InvStat in
                        0) InvStattxt="waiting" ;;
                        1) InvStattxt="normal" ;;
                        2) InvStattxt="fault" ;;
                esac

        # Ppv(W)
        Ppvt=$(hexdump -C -s $(calcpos 17) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
        Ppv=$(echo "ibase=16;obase=A;scale=1;$Ppvt/A" | bc)

        # Vpv1(V)
        Vpv1t=$(hexdump -C -s $(calcpos 21) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
        Vpv1=$(echo "ibase=16;obase=A;scale=1;$Vpv1t/A" | bc)

        # Ipv1(A)
        Ipv1t=$(hexdump -C -s $(calcpos 23) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
```

```
Ipv1=$(echo "ibase=16;obase=A;scale=1;$Ipv1t/A" | bc)

# Ppv1(W)
Ppv1t=$(hexdump -C -s $(calcpos 25) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
Ppv1=$(echo "ibase=16;obase=A;scale=1;$Ppv1t/A" | bc)

# Vpv2(V)
Vpv2t=$(hexdump -C -s $(calcpos 29) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Vpv2=$(echo "ibase=16;obase=A;scale=1;$Vpv2t/A" | bc)

# Ipv2(A)
Ipv2t=$(hexdump -C -s $(calcpos 31) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Ipv2=$(echo "ibase=16;obase=A;scale=1;$Ipv2t/A" | bc)

# Ppv2(W)
Ppv2t=$(hexdump -C -s $(calcpos 33) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
Ppv2=$(echo "ibase=16;obase=A;scale=1;$Ppv2t/A" | bc)

# Pac(W)
Pact=$(hexdump -C -s $(calcpos 37) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
Pac=$(echo "ibase=16;obase=A;scale=1;$Pact/A" | bc)

# Fac(Hz)
Fact=$(hexdump -C -s $(calcpos 41) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Fac=$(echo "ibase=16;obase=A;scale=2;$Fact/64" | bc)

# Vac1(V)
Vac1t=$(hexdump -C -s $(calcpos 43) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Vac1=$(echo "ibase=16;obase=A;scale=1;$Vac1t/A" | bc)

# Iac1(A)
Iac1t=$(hexdump -C -s $(calcpos 45) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Iac1=$(echo "ibase=16;obase=A;scale=1;$Iac1t/A" | bc)

# Pac1(W)
Pac1t=$(hexdump -C -s $(calcpos 47) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
Pac1=$(echo "ibase=16;obase=A;scale=1;$Pac1t/A" | bc)

# Vac2(V)
Vac2t=$(hexdump -C -s $(calcpos 51) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Vac2=$(echo "ibase=16;obase=A;scale=1;$Vac2t/A" | bc)

# Iac2(A)
Iac2t=$(hexdump -C -s $(calcpos 53) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Iac2=$(echo "ibase=16;obase=A;scale=1;$Iac2t/A" | bc)

# Pac2(W)
Pac2t=$(hexdump -C -s $(calcpos 55) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
Pac2=$(echo "ibase=16;obase=A;scale=1;$Pac2t/A" | bc)

# Vac3(V)
Vac3t=$(hexdump -C -s $(calcpos 59) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Vac3=$(echo "ibase=16;obase=A;scale=1;$Vac3t/A" | bc)

# Iac3(A)
Iac3t=$(hexdump -C -s $(calcpos 61) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Iac3=$(echo "ibase=16;obase=A;scale=1;$Iac3t/A" | bc)

# Pac3(W)
Pac3t=$(hexdump -C -s $(calcpos 63) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
Pac3=$(echo "ibase=16;obase=A;scale=1;$Pac3t/A" | bc)

# E_Today(kWh)  - pvoutput -> Energy
Etodayt=$(hexdump -C -s $(calcpos 67) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
Etoday=$(echo "ibase=16;obase=A;scale=2;$Etodayt*64" | bc)
Etodayk=$(echo "ibase=16;obase=A;scale=2;$Etodayt/A" | bc)

# E_Total (Wh)
Eallt=$(hexdump -C -s $(calcpos 71) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
Eall=$(echo "ibase=16;obase=A;scale=1;$Eallt/A" | bc)

# Tall (s) / TallH (h)
Tallt=$(hexdump -C -s $(calcpos 75) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
Tall=$(echo "ibase=16;obase=A;scale=2;$Tallt" | bc)
```

```
TallH=$(echo "$Tall/(60*60*2)" | bc)

# Tmp(C)
Tmpt=$(hexdump -C -s $(calcpos 79) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Tmp=$(echo "ibase=16;obase=A;scale=1;$Tmpt/A" | bc)

# ISO Fault (V)
ISOFt=$(hexdump -C -s $(calcpos 81) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
ISOF=$(echo "ibase=16;obase=A;scale=1;$ISOFt/A" | bc)

# GFCI Fault (mA)
GFCIFt=$(hexdump -C -s $(calcpos 83) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
GFCIF=$(echo "ibase=16;obase=A;scale=2;$GFCIFt/A" | bc)

# DCI Fault (A)
DCIFt=$(hexdump -C -s $(calcpos 85) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
DCIF=$(echo "ibase=16;obase=A;scale=2;$DCIFt/A" | bc)

# Vpvfault(V)
Vpvfaultt=$(hexdump -C -s $(calcpos 87) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Vpvfault=$(echo "ibase=16;obase=A;scale=1;$Vpvfaultt/A" | bc)

# Vacfault(V)
Vacfaultt=$(hexdump -C -s $(calcpos 89) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Vacfault=$(echo "ibase=16;obase=A;scale=1;$Vacfaultt/A" | bc)

# Facfault(Hz)
Facfaultt=$(hexdump -C -s $(calcpos 91) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Facfault=$(echo "ibase=16;obase=A;scale=2;$Facfaultt/64" | bc)

# Tmpfault(C)
Tmpfaultt=$(hexdump -C -s $(calcpos 93) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Tmpfault=$(echo "ibase=16;obase=A;scale=1;$Tmpfaultt/A" | bc)

# Faultcode(bits)
Faultcodet=$(hexdump -C -s $(calcpos 95) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Faultcode=$(echo "ibase=16;obase=2;$Faultcodet" | bc)

# IPMtemp(C)
IPMtempt=$(hexdump -C -s $(calcpos 97) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
IPMtemp=$(echo "ibase=16;obase=A;scale=1;$IPMtempt/A" | bc)

# Pbusvolt(V)
Pbusvoltt=$(hexdump -C -s $(calcpos 99) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Pbusvolt=$(echo "ibase=16;obase=A;scale=1;$Pbusvoltt/A" | bc)

# Nbusvolt(V)
Nbusvoltt=$(hexdump -C -s $(calcpos 101) -n 2 $fn | cut -c11-15 | tr a-f A-F | sed "s/ //g")
Nbusvolt=$(echo "ibase=16;obase=A;scale=1;$Nbusvoltt/A" | bc)

# Check step      - 103

# IPF             - 105

# RestCHK         - 107

# DeratingMode    - 109

# ?               - 111

# ?               - 113

# Epv1today (kWh)
Epv1todayt=$(hexdump -C -s $(calcpos 115) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
Epv1today=$(echo "ibase=16;obase=A;scale=2;$Epv1todayt/A" | bc)

# Epv1total (kWh)
Epv1totalt=$(hexdump -C -s $(calcpos 119) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
Epv1total=$(echo "ibase=16;obase=A;scale=2;$Epv1totalt/A" | bc)

# Epv2today (kWh)
Epv2todayt=$(hexdump -C -s $(calcpos 123) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
Epv2today=$(echo "ibase=16;obase=A;scale=2;$Epv2todayt/A" | bc)
```

```
        # Epv2total (kWh)
        Epv2totalt=$(hexdump -C -s $(calcpos 127) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
        Epv2total=$(echo "ibase=16;obase=A;scale=2;$Epv2totalt/A" | bc)

        # Epvtotal (kWh)
        Epvtotalt=$(hexdump -C -s $(calcpos 131) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
        Epvtotal=$(echo "ibase=16;obase=A;scale=2;$Epvtotalt/A" | bc)

        # Rac (Var)
        Ract=$(hexdump -C -s $(calcpos 135) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
        Rac=$(echo "ibase=16;obase=A;scale=2;$Ract*64" | bc)

        # E_Rac_today (Var)
        ERactodayt=$(hexdump -C -s $(calcpos 139) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
        ERactoday=$(echo "ibase=16;obase=A;scale=2;$ERactodayt*64" | bc)

        # E_Rac_total (Var)
        ERactotalt=$(hexdump -C -s $(calcpos 143) -n 4 $fn | cut -c11-21 | tr a-f A-F | sed "s/ //g")
        ERactotal=$(echo "ibase=16;obase=A;scale=2;$ERactotalt*64" | bc)

        #-------------------------------------------------------------------------
        # Write the details per record to the logfile
        #-------------------------------------------------------------------------
        printf "Growatt Inverter serial    (%s)"    $cs
        printf "                  Capture sample date : %s\n" $fd
        printf "Growatt Wifi Module serial (%s)"    $ah
        printf "                  Capture sample time : %s\n" $ft
        printf "Growatt Inverter status:    %s (%d)" $InvStattxt   $InvStat
        printf "                   Growatt temperature %6.1f C\n"  $Tmp
        printf "%87s\n" | tr ' ' -

        printf "%-11s %8.1f %-10s"      "E_Today"    $Etodayk "kWh"
        printf "%-11s %8.1f %-10s"      "E_Total"    $Eall    "kWh"
        printf "%-11s %8.1f %-10s\n"    "Total time" $TallH   "hrs"
        printf "%87s\n" | tr ' ' -

        printf "%-11s %8.1f %-10s"      "Ppv" $Ppv " W"
        printf "%-11s %8.1f %-10s"      "Pac" $Pac " W"
        printf "%-11s %8.2f %-10s\n"    "Fac" $Fac "Hz"
        printf "%87s\n" | tr ' ' -

        printf "%-11s %8.1f %-10s"      "Vpv1" $Vpv1 " V"
        printf "%-11s %8.1f %-10s\n"    "Vpv2" $Vpv2 " V"
        printf "%-11s %8.1f %-10s"      "Ipv1" $Ipv1 " A"
        printf "%-11s %8.1f %-10s\n"    "Ipv2" $Ipv2 " A"
        printf "%-11s %8.1f %-10s"      "Ppv1" $Ppv1 " W"
        printf "%-11s %8.1f %-10s\n"    "Ppv2" $Ppv2 " W"
        printf "%87s\n" | tr ' ' -

        printf "%-11s %8.1f %-10s"      "Vac1" $Vac1 " V"
        printf "%-11s %8.1f %-10s"      "Vac2" $Vac2 " V"
        printf "%-11s %8.1f %-10s\n"    "Vac3" $Vac3 " V"
        printf "%-11s %8.1f %-10s"      "Iac1" $Iac1 " A"
        printf "%-11s %8.1f %-10s"      "Iac2" $Iac2 " A"
        printf "%-11s %8.1f %-10s\n"    "Iac2" $Iac3 " A"
        printf "%-11s %8.1f %-10s"      "Pac1" $Pac1 " W"
        printf "%-11s %8.1f %-10s"      "Pac2" $Pac2 " W"
        printf "%-11s %8.1f %-10s\n"    "Pac2" $Pac3 " W"
        printf "%87s\n" | tr ' ' -

        printf "%-11s %8.1f %-10s\n"    "Epvtotal"  $Epvtotal  "kW"
        printf "%-11s %8.1f %-10s"      "Epv1today" $Epv1today "kW"
        printf "%-11s %8.1f %-10s\n"    "Epv2today" $Epv2today "kW"
        printf "%-11s %8.1f %-10s"      "Epv1total" $Epv1total "kW"
        printf "%-11s %8.1f %-10s\n"    "Epv2total" $Epv2total "kW"
        printf "%87s\n" | tr ' ' -

        printf "%-11s %8.1f %-10s"      "ISO Fault"  $ISOF      " V"
        printf "%-11s %8.1f %-10s"      "Vpvfault"   $Vpvfault  " V"
        printf "%-11s %8.1f %-10s\n"    "Tempfault"  $Tmpfault  " C"
        printf "%-11s %8.1f %-10s"      "GFCI Fault" $GFCIF     "mA"
        printf "%-11s %8.1f %-10s"      "Vacfault"   $Vacfault  " V"
        printf "%-11s %11s\n"           "Faultcode"  $Faultcode
        printf "%-11s %8.1f %-10s"      "DCI Fault"  $DCIF      " A"
```

```
        printf "%-11s %8.2f %-10s\n"    "Facfault"   $Facfault  "Hz"
        printf "%87s\n" | tr ' ' -

        printf "%-11s %8.1f %-10s"      "IPMtemp"     $IPMtemp     " C"
        printf "%-11s %8.1f %-10s\n"    "Rac"         $Rac         "Var"
        printf "%-11s %8.1f %-10s"      "Pbusvolt"    $Pbusvolt    " V"
        printf "%-11s %8.1f %-10s\n"    "E_Rac_today" $ERactoday   "Var"
        printf "%-11s %8.1f %-10s"      "Nbusvolt"    $Nbusvolt    " V"
        printf "%-11s %8.1f %-10s\n"    "E_Rac_total" $ERactotal   "Var"
        printf "%87s\n" | tr ' ' -

else
        #-------------------------------------------------------------------------
        # Write error message if the file is of an incorrect length
        #-------------------------------------------------------------------------
        printf "\nERROR: The file does not have the expected length...\n\n"
        printf "The length needs to be between %s and %s characters,\n" $LOWERLIMIT $UPPERLIMIT
        printf "but is currently %s characters.\n\n" $fs
fi
```

5.2 Processing script: `process_growatt_pvoutput.sh`

This script is created to capture the minimum required data from a capture record that is needed to upload to `pvoutput.org`. The minimum data required are the date (20140607), time (09:51), energy generated in Watts – E_Today/E_Total (8800) and current power in Watts – Pac (841.9).

The script includes error handling if the data records are incorrect, but also when the upload fails.

5.2.1 User variables

Ensure that the user variables are changed according to your local setup:

```
PVOUTPUTKEY="<your PVOutput API key>"
PVOUTPUTSID="<your PVOutput SID>"
GROWATTSERIAL="<your Growatt inverter serial number>"
```

These two options are added to indicate if Vpv1 or Vpv1/Vpv2 need to be uploaded to PVOutput. The default options are:

```
PVOUTPUTVPV1=NO
PVOUTPUTVPV2=NO
```

The following table shows the possibilities:

	PVOUTPUTVPV1	**PVOUTPUTVPV2**
Do not upload Vpv1/Vpv2 data	NO	NO
Upload Vpv1 data only	YES	NO
Upload Vpv1 and Vpv2 data	YES	YES

This option is used to change which power generation variable is uploaded (E_Today or E_Total). This flag adds support for the PVOutput c1 option. With this cumulative flag it is possible to use the E_Total generation number instead of the E_Today number. This can be helpful if the inverter is restarted during the day, caused by insufficient power (e.g. bad weather) or restarts during failures. Normally the E_Today counter starts at zero again, whereas E_Today keeps counting.

```
PVCUMFLAG=1
```

	PVCUMFLAG
Upload E_Today (default)	0
Upload E_Total	1

The `DESTINATION` variable needs to be set to the fully qualified domain name (FQDN) of the Growatt server to which the data is forwarded. The default value is `server.growatt.com`. Normally the Raspberry Pi has `eth0` as value for `INTERFACE`. If your Raspberry Pi has a different interface name then also change it accordingly in the file above to ensure capturing from the right ethernet device.

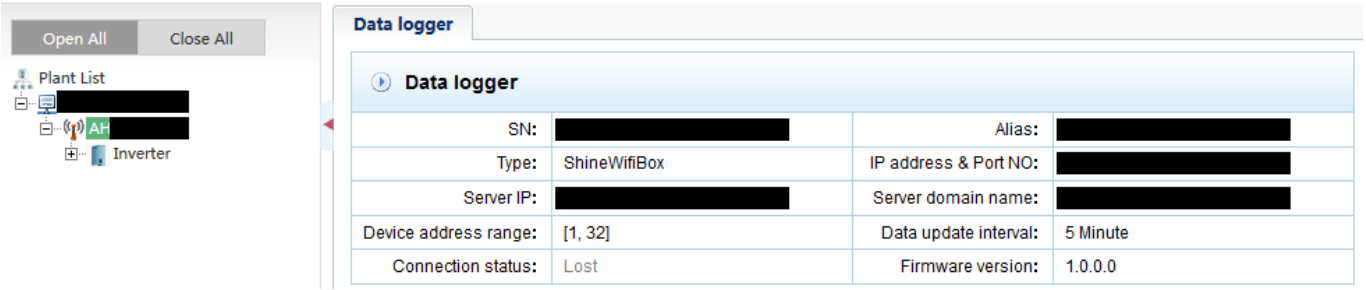
```
DESTINATION=server.growatt.com
INTERFACE=eth0
```

This variable is used to take into account the different data positions for later firmwares of the Growatt Wifi Module. At this moment there are two known versions (which are both supported).

```
GROWATTMODULEVER=1.0.0.0
```

	GROWATTMODULEVER
First Growatt Wifi Module (default)	1.0.0.0
Later release of Growatt Wifi Module	2.0.0.0
Latest release of Growatt Wifi Module	3.0.0.0

The firmware version can be found in the http://server.growatt.com interface. Navigate down the tree on the left from “Plant List” to the “Plant” to the serial number of the data logger (starting with AH). The Firmware version is shown on the bottom right.



5.2.2 Version 2.3

```
#!/bin/sh
#-------------------------------------------------------------------------
# Name      : process_growatt_pvoutput.sh
#
# Function  : Read Growatt Wifi Module records to server.growatt.com,
#             extract relevant data and submit to pvoutput.org account.
#
# Date      : 09-07-2015
# Author    : S. Plug
#
# Version   : 2.3 Added support for Growatt Wifi Module 3.0.0.0 firmware.
#                 Added solution to accomodate for chaning ip-address of
#                 the server.growatt.com destination.
#             2.2 Set cumulative flag (PVCUMFLAG) default to 1 (enabled)
#                 Added tcpdump logic improvement from Menno Norden to
#                 avoid promiscuous mode messages in the syslog (-p) and
#                 set the recording to 300 seconds with a check if it is
#                 still running. This avoids data gaps.
#             2.1 Added PVOutput c1 option (cumulative flag)
#                 Added data postion correction (calcpos) based upon the
#                 Growatt Wifi Module firmware version.
#                 Removed the fixed path for curl, now it is assumed it
#                 can be found in the $PATH like the other commands.
#                 Changed logfile messages and formatting.
#                 Changed file names during processing to bring clarity.
#                 Excluded files not preprocessed from processing routine.
#             2.0 Different checking/splitting of input files. Added
#                 support for Vpv1 and Vpv2 values.
#             1.1 Added offset calculation to accomodate for different
#                 inverters/data records. Changed value labels to match
#                 Growatt values.
#
#-------------------------------------------------------------------------

# Uncomment to have debugging information
# set -x

#-------------------------------------------------------------------------
# Declaration of user variables
#-------------------------------------------------------------------------
PVOUTPUTKEY="<your PVOutput API key>"
PVOUTPUTSID="<your PVOutput SID>"
GROWATTSERIAL="<your Growatt inverter serial number>"
PVOUTPUTVPV1=NO
PVOUTPUTVPV2=NO
PVCUMFLAG=1
GROWATTMODULEVER=1.0.0.0

#-------------------------------------------------------------------------
# Declaration interfaces and Growatt server destination
#-------------------------------------------------------------------------
DESTINATION=server.growatt.com
INTERFACE=eth0

#-------------------------------------------------------------------------
# Declaration of script variables
#-------------------------------------------------------------------------
PVOUTPUTURL="http://pvoutput.org/service/r2/addstatus.jsp"
CURLTIMEOUT=60
PVBASEDIR=/home/pvoutput
PVBASEDIR=/home/pvoutput/
PVINDIR=$PVBASEDIR/input
PVOUTDIR=$PVBASEDIR/processed
PVLOGDIR=$PVBASEDIR/logs
PVLOGFILE="$PVLOGDIR/growatt_process.log"
PVTMPFILE="$PVBASEDIR/tmp/growattsize"
PVCAPFILE="growatt_*.cap"
PVPREFILE="growatt_*.preproc"
```

```
LOWERLIMIT=300
DIVIDER="--------------------------------------------------------------------------------------------"

#---------------------------------------------------------------------------
# Function to accomodate for the changing ip-address of the destination
# server: http://server.growatt.com
#---------------------------------------------------------------------------
destip()
{
        #---------------------------------------------------------------------------
        # Determine the IP address ${DESTIP} from ${DESTINATION} and determine
        # the IP address used in the current iptables rule.
        #---------------------------------------------------------------------------
        DESTIP=$(host ${DESTINATION} | cut -f4 -d' ')
        CURRIP=$(iptables -t nat -S PREROUTING | grep  "\-\-dport 5279" | \
                        rev | cut -f 1 -d ' ' | rev | cut -f 1 -d ':')

        #---------------------------------------------------------------------------
        # Provide the definitions of the PREROUTING and POSTROUTING iptable rules.
        #---------------------------------------------------------------------------
        IPTACO="iptables -t nat"
        IPTAR0="PREROUTING -p tcp --dport 5279 -j DNAT --to-destination ${CURRIP}:5279"
        IPTAR1="PREROUTING -p tcp --dport 5279 -j DNAT --to-destination ${DESTIP}:5279"
        IPTAR2="POSTROUTING -j MASQUERADE"

        #---------------------------------------------------------------------------
        # PREROUTING rule does not exists and will be added.
        #---------------------------------------------------------------------------
        if [ "${CURRIP}" = "" ]; then
                ${IPTACO} -A ${IPTAR1}
                echo "iptables added:     ${IPTAR1}" >> $PVLOGFILE

        #---------------------------------------------------------------------------
        # Check for the existance and correctness of the PREROUTING rule and add
        # or change it if needed.
        #---------------------------------------------------------------------------
        elif [ "${CURRIP}" != "${DESTIP}" ]; then
                ${IPTACO} -C ${IPTAR0}
                if [ ${?} -eq 0 ]; then
                        ${IPTACO} -D ${IPTAR0}
                        echo "iptables removed:   ${IPTAR0}" >> $PVLOGFILE
                fi

                ${IPTACO} -A ${IPTAR1}
                echo "iptables added:     ${IPTAR1}" >> $PVLOGFILE

        #---------------------------------------------------------------------------
        # No need to change the PREROUTING rule which correctly exists already.
        #---------------------------------------------------------------------------
        elif [ "${CURRIP}" = "${DESTIP}" ]; then
                echo "iptables unchanged: ${IPTAR1}" >> $PVLOGFILE
        fi

        #---------------------------------------------------------------------------
        # Check for the existance of the POSTROUTING rule and add it if needed.
        #---------------------------------------------------------------------------
        ${IPTACO} -C ${IPTAR2}
        if [ ${?} -eq 0 ]; then
                echo "iptables unchanged: ${IPTAR2}" >> $PVLOGFILE
        else
                ${IPTACO} -A ${IPTAR2}
                echo "iptables added:     ${IPTAR2}" >> $PVLOGFILE
        fi
}

#---------------------------------------------------------------------------
# Function to calculate register position based on $OFFSET and parameter.
# There are different Growatt Wifi Module firmware versions.
# The difference between the positions of the position of GROWATTSERIAL
# and the position of the data is +6 positions when 2.0.0.0 firmware or
# 3.0.0.0 firmware is # used (compared to 1.0.0.0 firmware).
#---------------------------------------------------------------------------
calcpos()
{
```

```
        case "$GROWATTMODULEVER" in
        "1.0.0.0")
                OFFSETCORRECT=0
        ;;
        "2.0.0.0")
                OFFSETCORRECT=6
        ;;
        "3.0.0.0")
                OFFSETCORRECT=6
        ;;
        esac

        echo $(echo $OFFSET + $1 + $OFFSETCORRECT | bc)
}

#-------------------------------------------------------------------------
# Function to check the position of a string in the HEX file
#-------------------------------------------------------------------------
offsetcheck()
{
    echo $(grep -obUaP \
         $(echo -n "$1" | od -A n -t x1 | sed "s/ /\\\x/g") $fn \
         | cut -d ":" -f 1)
}

#-------------------------------------------------------------------------
# Write start of new processing to logfile
#-------------------------------------------------------------------------
echo ${DIVIDER} >> ${PVLOGFILE}
date >> ${PVLOGFILE}
echo ${DIVIDER} >> ${PVLOGFILE}

#-------------------------------------------------------------------------
# Determine if the IP address of growatt.server.com changed
#-------------------------------------------------------------------------
destip

#-------------------------------------------------------------------------
# Capture the Growatt Traffic (stop when 1 packet captured or 5 minutes past)
#-------------------------------------------------------------------------
PID=$(pidof tcpdump)
if [ -n "$PID" ]; then
      kill $PID
      printf "tcpdump killed \n" >> $PVLOGFILE
fi
/usr/sbin/tcpdump -p -i $INTERFACE -nn -G 60 -s 300 -c 1 -w \
                        $PVINDIR/growatt_%Y%m%d_%H:%M.cap \
                        greater 220 and tcp and less 500 and \
                        dst $DESTIP >/dev/null 2>&1

#-------------------------------------------------------------------------
# Remove Growatt data capture files with incorrect length.
# DO NOT remove files with 0 length as they still capture data
# If no capture files are found then write a message in the log and exit.
#-------------------------------------------------------------------------
for fn in $PVINDIR/$PVCAPFILE
do
        #-----------------------------------------------------------------------
        # Determine if files ($PVCAPFILE) exist in the input directory ($PVINDIR)
        #-----------------------------------------------------------------------
        if [ ! -f "$fn" ]; then
                printf "%-50s\n" "No capture in input directory. Processing ended." \
                       >> ${PVLOGFILE}
                exit 0
        else
                fs=$(stat --format=%s "$fn")
                pd=$(echo $fn | cut -d "_" -f 4)

                # Split capture files in 1 file per packet in $PVTMPFILE and
                # move original capture packet to $PVOUTDIR with .split extension
                if [ "$fs" -gt $LOWERLIMIT ] && [ "$pd" = "" ]; then
                        editcap -c 1 $fn $PVTMPFILE
                        mv $fn ${PVOUTDIR}/${fn##*/}.split
                        printf "%-50s : %s\n" \
```

```
                                "Capture file split, moved to ${PVOUTDIR##*/} dir." \
                                ${fn##*/}.split >> ${PVLOGFILE}
                fi

                # Move small non-empty files to $PVOUTDIR
                if [ "$fs" -gt 0 -a "$fs" -lt $LOWERLIMIT ]; then
                        mv $fn ${PVOUTDIR}/${fn##*/}.smallsize
                        printf "%-50s : %s\n" \
                                "Capture file < ${LOWERLIMIT} chars, moved to ${PVOUTDIR##*/} dir." \
                                ${fn##*/}.smallsize >> ${PVLOGFILE}
                fi
        fi
done

#------------------------------------------------------------------------
# Preprocess the input files
#------------------------------------------------------------------------
for fn in $PVTMPFILE*
do
        if [ -f "$fn" ]; then
                fnnew=$(printf "growatt_%s_%s:%s_%s.preproc" \
                      $(echo $fn | cut -d "_" -f 3 | cut -c1-8)   \
                      $(echo $fn | cut -d "_" -f 3 | cut -c9-10)  \
                      $(echo $fn | cut -d "_" -f 3 | cut -c11-12) \
                      $(echo $fn | cut -d "_" -f 3 | cut -c13-14))

                #------------------------------------------------------------------------------
                # Use $GROWATTSERIAL to find the reference point to be used as offset
                #------------------------------------------------------------------------------
                STRING1=$(offsetcheck "Inverter")       # Exclude equal sized packet (no data)
                STRING2=$(offsetcheck "5279")           # Exclude (re)logon packet (portnumber)
                GSERIAL=$(offsetcheck "$GROWATTSERIAL") # Exclude non-match serial packet

                if [ "$STRING1" = "" ] && [ "$STRING2" = "" ] && [ "$GSERIAL" != "" ] ; then
                        mv $fn $PVINDIR/${fnnew##*/}
                        printf "%-50s : %s\n" \
                                "Valid file moved to ${PVINDIR##*/} dir." \
                                ${fnnew##*/} >> ${PVLOGFILE}
                else
                        printf "%-50s : %s\n" \
                                "Invalid file removed" \
                                ${fn##*/} >> ${PVLOGFILE}
                        rm $fn
                fi
        fi
done

#------------------------------------------------------------------------
# Process valid (already preprocessed) files
#------------------------------------------------------------------------
for fn in $PVINDIR/$PVPREFILE
do
        #------------------------------------------------------------------------------
        # Determine if files ($PVCAPFILE) exist in the input directory ($PVINDIR)
        #------------------------------------------------------------------------------
        if [ ! -f "$fn" ]; then
                printf "%-50s\n" "No input files to be processed. Processing ended." \
                       >> ${PVLOGFILE}
                exit 0
        fi

        #------------------------------------------------------------------------------
        # Fill the variables for Date (fd) and Time (ft)
        #------------------------------------------------------------------------------
        fd=$(echo $fn | cut -d_ -f2)
        ft=$(echo $fn | cut -d_ -f3 | cut -c1-5)

        #------------------------------------------------------------------------------
        # Use $GROWATTSERIAL to find the reference point to be used as offset
        # Also exclude empty files from processing
        #------------------------------------------------------------------------------
        OFFSET=$(offsetcheck "$GROWATTSERIAL")
        fs=$(stat --format=%s "$fn")
```

```
        if [ "$OFFSET" = "" ] && [ $fs -ne 0 ]; then
                printf "%s;%s;"    $fd $ft   >> ${PVLOGFILE}
                printf "%6s;%6s;"  "NA" "NA" >> ${PVLOGFILE}
                printf "%-50s : %s\n" "File removed. Serial number ${GROWATTSERIAL} not found" \
                        ${fn##*/} >> ${PVLOGFILE}
                        rm $fn

        elif [ $fs -ne 0 ]; then
                #--------------------------------------------------------------------------
                # Extract Growatt Inverter information
                #--------------------------------------------------------------------------

                # Use E_Total or E_Today depending on the users preference (PVCUMFLAG)
                if [ "$PVCUMFLAG" = 1 ]; then
                        # E_Total (Wh) - pvoutput -> Energy (v1) if c1=1
                        ETt=$(hexdump -C -s $(calcpos 71) -n 4 $fn \
                             | cut -c11-21 | tr a-f A-F | sed "s/ //g")
                        ET=$(echo "ibase=16;obase=A;scale=2;$ETt*64" | bc)
                        PVENERGY=$ET
                else
                        # E_Today(Wh)  - pvoutput -> Energy (v1) if c1=0
                        ETdt=$(hexdump -C -s $(calcpos 67) -n 4 $fn \
                             | cut -c11-21 | tr a-f A-F | sed "s/ //g")
                        ETd=$(echo "ibase=16;obase=A;scale=2;$ETdt*64" | bc)
                        PVCUMFLAG=0
                        PVENERGY=$ETd
                fi

                # Pac(W)        - pvoutput -> Power (v2)
                Pact=$(hexdump -C -s $(calcpos 37) -n 4 $fn \
                     | cut -c11-21 | tr a-f A-F | sed "s/ //g")
                Pac=$(echo "ibase=16;obase=A;scale=1;$Pact/A" | bc)

                # Vpv1(V)
                if [ "$PVOUTPUTVPV1" = "YES" ]; then
                        Vpv1t=$(hexdump -C -s $(calcpos 21) -n 2 $fn \
                              | cut -c11-15 | tr a-f A-F | sed "s/ //g")
                        Vpv1=$(echo "ibase=16;obase=A;scale=1;$Vpv1t/A" | bc)
                fi

                # Vpv2(V)
                if [ "$PVOUTPUTVPV2" = "YES" ]; then
                        Vpv2t=$(hexdump -C -s $(calcpos 29) -n 2 $fn \
                      | cut -c11-15 | tr a-f A-F | sed "s/ //g")
                        Vpv2=$(echo "ibase=16;obase=A;scale=1;$Vpv2t/A" | bc)
                fi

                #--------------------------------------------------------------------------
                # Write the details per record to the logfile
                #--------------------------------------------------------------------------
                printf "%s;%s;"      $fd       $ft  >> ${PVLOGFILE}
                printf "%10u;%7.1f;" $PVENERGY $Pac >> ${PVLOGFILE}
                printf "%7.1f;"      $Vpv1          >> ${PVLOGFILE}
                printf "%7.1f;"      $Vpv2          >> ${PVLOGFILE}

                #--------------------------------------------------------------------------
                # Submit the data to pvoutput.org servers and capture the result
                #--------------------------------------------------------------------------

                if [ "$PVOUTPUTVPV1" = "YES" ] && [ "$PVOUTPUTVPV2" = "YES" ]; then

                        result=$(curl -s -S --max-time $CURLTIMEOUT \
                                -d "d=$fd"    -d "t=$ft"    -d "v1=$PVENERGY" -d "v2=$Pac" \
                                -d "v6=$Vpv1" -d "v7=$Vpv2" \
                                -d "c1=$PVCUMFLAG" \
                                -H "X-Pvoutput-Apikey: $PVOUTPUTKEY" \
                                -H "X-Pvoutput-SystemId: $PVOUTPUTSID" $PVOUTPUTURL 2>&1)

                elif [ "$PVOUTPUTVPV1" = "YES" ] && [ "$PVOUTPUTVPV2" != "YES" ]; then

                        result=$(curl -s -S --max-time $CURLTIMEOUT \
                                -d "d=$fd"    -d "t=$ft"    -d "v1=$PVENERGY" -d "v2=$Pac" \
                                -d "v6=$Vpv1"  \
                                -d "c1=$PVCUMFLAG" \
```

```
                                -H "X-Pvoutput-Apikey: $PVOUTPUTKEY" \
                                -H "X-Pvoutput-SystemId: $PVOUTPUTSID" $PVOUTPUTURL 2>&1)

                elif [ "$PVOUTPUTVPV1" != "YES" ]; then

                        result=$(curl -s -S --max-time $CURLTIMEOUT \
                                -d "d=$fd" -d "t=$ft" -d "v1=$PVENERGY" -d "v2=$Pac" \
                                -d "c1=$PVCUMFLAG" \
                                -H "X-Pvoutput-Apikey: $PVOUTPUTKEY" \
                                -H "X-Pvoutput-SystemId: $PVOUTPUTSID" $PVOUTPUTURL 2>&1)
                fi

                echo $result >> ${PVLOGFILE}

                #--------------------------------------------------------------------------
                # Veify succesful upload and archive the files to processed directory
                #--------------------------------------------------------------------------
                rescode200=$(echo $result | grep "OK 200: Added Status" | wc -l)
                rescode400=$(echo $result | grep "Bad request 400" | wc -l)
                rescode403=$(echo $result | grep "Forbidden 403: Exceeded" | wc -l)

                # To many uploads within one hour
                if [ "$rescode403" -eq 1 ]; then
                        exit 1
                fi

                # Upload failed
                if [ "$rescode400" -eq 1 ]; then
                        mv $fn ${PVOUTDIR}/${fn##*/}.badupload
                fi

                # Upload succeeded
                if [ "$rescode200" -eq 1 ]; then
                        mv $fn ${PVOUTDIR}/${fn##*/}.ok
                        printf "%-50s : %s\n" \
                               "Processed file, moved to ${PVOUTDIR##*/} dir." \
                               ${fn##*/}.ok >> ${PVLOGFILE}
                fi

                #--------------------------------------------------------------------------
                # Reset the global variables for the next run
                #--------------------------------------------------------------------------
                result=""
                rescode200=0
                rescode400=0
                rescode403=0
        fi
done
```