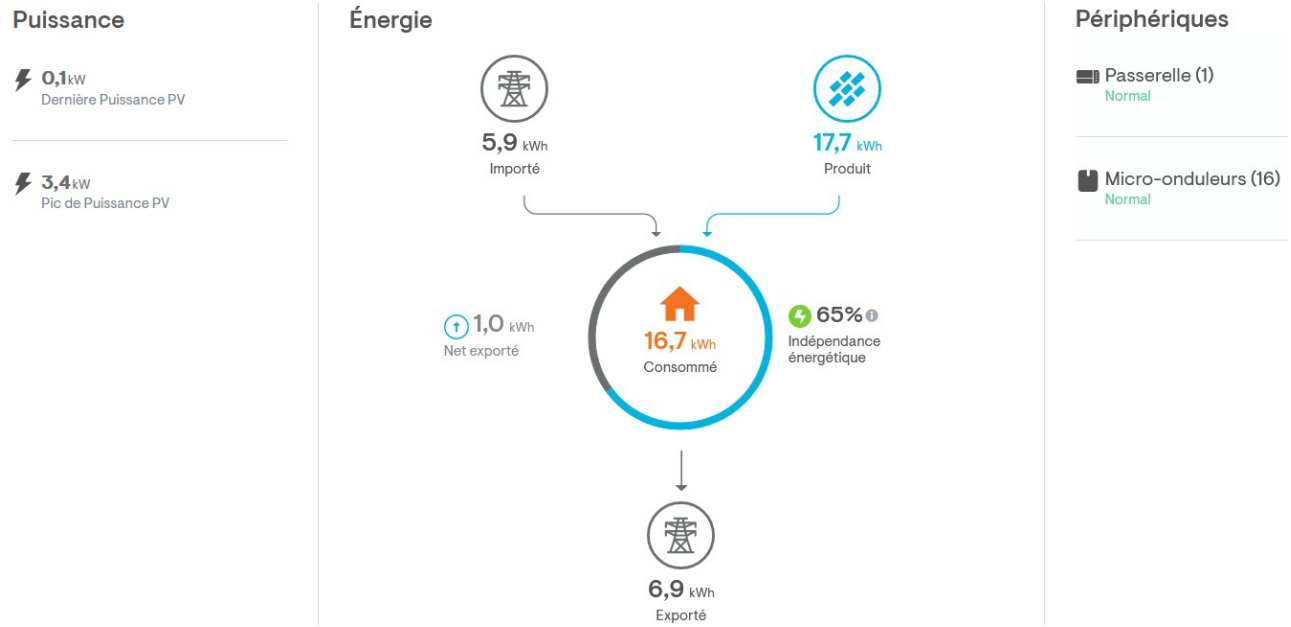
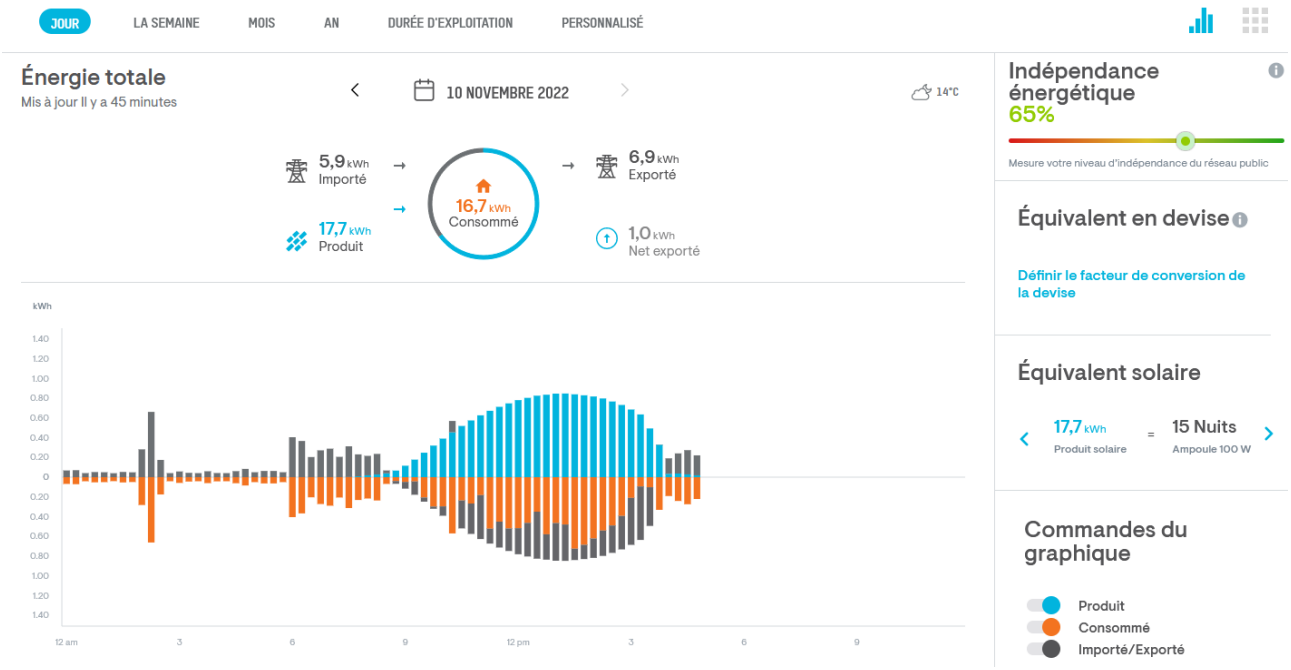
Panneaux PhotoVoltaïques : mon chauffage gratuit

J'adore les trucs informatiques de geeks.. mais encore plus quand ça sert à la vraie vie :-)

Propriétaire d'une maison depuis le début de l'année, pas mal consommatrice d'électricité, j'ai vite voulu opter pour des panneaux solaires en autoconsommation.
En Août je fais poser 6Kwc sur le toit, une installation assez classique comme on en voit beaucoup, notamment avec une passerelle de type ENPHASE.
La passerelle doit être reliée au wifi (ou réseau) interne pour fonctionner, les installateurs me paramètrent la passerelle, me donnent un lien vers un site et une appli ENPHASE et… c'est tout.
Je ne leur en veux pas c'est pas leur métier, mais c'est un peu faible pour maîtriser son installation, on accède juste à une appli sur son smartphone pour voir le détail des consommations heure par heure.

L'appli et le site ressemblent à ça :





C'est joli, bien pour les statistiques et le suivi régulier, mais j'en veux un peu plus.

Voici donc la suite.

1ère étape, jeter un œil à la passerelle.
Une petite recherche sur mon réseau interne me donne son IP : 192.168.0.24. Sur ma Freebox je lui force cette IP en bail DHCP statique, histoire de pouvoir toujours l'adresser facilement.

Puis j'essaye un truc tout bête : qu'est-ce qu'il y a derrière cette IP ?
Je lance mon navigateur préféré, je tape l'adresse ***http://192.168.0.24*** et.. bingo, une page web :

C'est un peu plus cool ça, en « direct live » avec cette page j'ai :

- ma production photovoltaïque,
- la consommation de la maison
- et par différence, puissance nette, ce que j'importe du réseau ENEDIS pour complément ou ce que j'exporte vers le réseau en revente.

Pas mal, j'ai déjà un outil utile à consulter pour, par exemple, lancer manuellement mes appareils consommateurs d'énergie quand j'ai de la disponibilité d'électricité gratuite :

- une machine à laver ou un sèche linge
- un lave-vaisselle
- un cumulus d'eau chaude en mode forcé
- ..

Ça c'est quand je suis à la maison, mais il y a un autre sujet qui m'intéresserait : pouvoir lancer automatiquement un chauffage chez moi quand je n'y suis pas et que mes panneaux produisent.
Il y a une petite contrainte donc : comment allumer mon chauffage quand les panneaux produisent suffisamment, et comment l'éteindre quand il n'y en a plus assez (pluie, passage de nuages, fin de journée…) ?

Le chauffage de la pièce de vie consomme environ 2000 Watt, l'idéal serait donc de connaître en temps réel si la différence entre ce que je produis et ce que ma maison consomme me laisse assez de puissance pour couvrir 2000 Watt.
Bien entendu je veux automatiser le processus, c'est donc quelque chose que je piloterai avec mon raspberry sous Raspbian.

1ère tentative : récupérer les données de la page de la passerelle ENPHASE.. je tente avec cURL :

curl http://192.168.0.24/home

Le résultat n'est pas bon, il y a du javascript, rien n'est interprété par curl, je n'ai pas les données.
Je fais quelques recherches sur le net et je découvre que la passerelle ENPHASE met aussi à disposition des pages internes pour récupérer des données au format json.
Très très utile ça.. je teste les 3 liens json que j'ai trouvés chez moi dans mon navigateur :

http://192.168.0.24/home.json
http://192.168.0.24/production.json
http://192.168.0.24/inventory.json

Les résultats sont tous utiles et détaillés mais c'est celui-ci ***http://192.168.0.24/production.json*** *qui* m'affiche des données à exploiter puisque j'y trouve la production et la consommation en direct.

Retour donc à curl :

curl http://192.168.0.24/production.json

C'est déjà bien mais le format est brut, tout est écrit sur une seule ligne... pas pratique.
Une nouvelle recherche sur le net me renvoie sur l'utilisation d'un outil de filtre et d'interprétation du format json pour Linux : **jq**

J'installe et je teste d'envoyer le résultat à **jq** :

apt install jq
curl http://192.168.0.24/production.json | jq

Ca me donne un truc comme ça :


```
{
  "production": [
    {
      "type": "inverters",
      "activeCount": 16,
      "readingTime": 1668103077,
      "wNow": 0,
```

```
    "whLifetime": 2038223
   },
   {
    "type": "eim",
    "activeCount": 1,
    "measurementType": "production",
    "readingTime": 1668103417,
    "wNow": 1221.637,
    "whLifetime": 2058702.742,
    ...
   }
  ],
  "consumption": [
   {
    "type": "eim",
    "activeCount": 1,
    "measurementType": "total-consumption",
    "readingTime": 1668103417,
    "wNow": 1421.959,
    "whLifetime": 1428028.156,
    ...
   },
```


OK on avance, les deux lignes qui m'intéressent sont bien là :

"wNow": 1221.637,
"wNow": 1421.959,

La 1ere, dans la section « production » me donne ce que je produis : **1221.637 watt**
La 2eme, dans la section «consumption» me donne ce que ma maison consomme : **1421.959 watt**

jq permet de filtrer et de sortir uniquement un champs donné en paramètre. Au final j'en déduis les deux syntaxe curl + jq pour avoir précisément ce que je veux :

curl -s 192.168.0.24/production.json | jq ".production[1].wNow"

et

curl -s 192.168.0.24/production.json | jq ".consumption[0].wNow"

(⇒ A noter le pointage du champs par section dans le json [0] et [1])

Nickel, je me fais un premier petit script bash Linux pour exploiter ça :

```
#!/bin/bash

## recup productiopn temps reel
prod=`curl -s 192.168.0.24/production.json | jq ".production[1].wNow"`
## recup consommation temps reel
conso=`curl -s 192.168.0.24/production.json | jq ".consumption[0].wNow"`
## delta
delta=`echo "$prod-$conso" | bc`

echo ""
echo "production : "$prod
echo "consommation : "$conso
echo "delta actuel : "$delta
echo ""

if (( $(echo "$delta < 0" |bc -l) ))
then
      echo "On consomme :-( "
else
      echo "On produit :-) !!! "
fi
echo ""
```

Dans l'ordre le script :

- récupère les valeurs de consommation et production dans deux variable « prod » et « conso »
- calcule le delta (prod – conso) dans la variable « delta » a l'aide de la commande bc
- affiche les valeurs
- affiche un message selon qu'on consomme ou qu'on produit (delta positif ou négatif)

Exemple de retour :

production : -0.766
consommation : 1333.073
delta actuel : -1333.839
On consomme :-(

ou

production : 1254,766
consommation : 233.568
delta actuel : 1021.198
On produit :-) !!!

Ça fonctionne et c'est fun. Maintenant il faut coupler ça au déclenchement ou non d'un chauffage.

Pour cela j'utilise une prise connectée de type CHACON que je pilote avec mon raspberry et un module radio RF433. Je vous renvois vers mon autre tuto à ce sujet :
http://superlemming.free.fr/cmsmadesimple/uploads/tutoriaux/tutoriel-prises-commandees.pdf

On retiendra juste pour l'exemple que pour allumer ma prise je lance :

/root/433Utils/Rpi_utils/codesend 1381717

et pour l'éteindre :

/root/433Utils/Rpi_utils/codesend 1381718

Je branche bien entendu mon chauffage à la prise connectée, et crée mon nouveau script :

```
#!/bin/bash

cd /root/scripts/pv

## recup productiopn temps reel
prod=`curl -s 192.168.0.24/production.json | jq ".production[1].wNow"`
## recup consommation temps reel
conso=`curl -s 192.168.0.24/production.json | jq ".consumption[0].wNow"`
## delta
delta=`echo "$prod-$conso" | bc`

if [ -f ./CHAUFFAGE_ON ]
then
   if (( $(echo "$delta < 0" |bc -l) ))
   then
      /root/433Utils/Rpi_utils/codesend 1381718
      rm ./CHAUFFAGE_ON
   fi
else
   if (( $(echo "$delta > 2000" |bc -l) ))
   then
      /root/433Utils/Rpi_utils/codesend 1381717
      touch ./CHAUFFAGE_ON
   fi
fi
```

Que fait ce nouveau script, explications :

- On pousse les valeurs de consommation, production et delta (production – consommation) dans des variables
- On teste si le chauffage n'est pas déjà allumé à l'aide de la simple présence ou non d'un fichier nommé CHAUFAGE_ON
- Si le chauffage est déjà allumé on teste si le delta de production-conso n'est pas négatif (< 0) ce qui signifierait que <u>la production ne couvre plus la consommation</u> ⇒ Si c'est le cas on arrête le chauffage et on supprime le fichier de test CHAUFFAGE_ON
- Si le chauffage n'est pas déjà allumé on teste si le delta de production-conso est suffisant pour couvrir le chauffage (>2000), ce qui signifierait que <u>la production pourrait couvrir en plus la consommation du chauffage</u> ⇒ Si c'est le cas on allume le chauffage et on crée le fichier de test CHAUFFAGE_ON

L'utilisation d'un fichier de test est indispensable, le piège aurait été de simplement tester le delta > 2000… mais avec un delta suffisant si on allume le chauffage, il se met à consomme et donc le delta n'est plus de 2000, ce qui va générer des allumages/extinctions en boucle :-)

Ici les allumages se font en mode « gratuit », avec aucune tolérance de sur-consommation, mais on peut aussi faire varier les deux tests de « delta » si on accepte de consommer un petit peu d'électricité venant du réseau.
Par exemple on peut choisir d'éteindre le chauffage uniquement si on sur-consomme/importe plus de 500 watt du réseau :

```
if (( $(echo "$delta < -500" |bc -l) ))
```

.. ou l'allumer même s'il n'y a que 1500 watt de disponible pour notre chauffage de 2000 watt :

```
if (( $(echo "$delta > 1500" |bc -l) ))
```

Il n’y a plus qu’à mettre ça dans un crontab pour le lancer toutes les 1 minutes (par exemple), entre 10H et 17H l’hiver quand on sait que la production est susceptible de suffire :

/1 10-17 * * * /root/scripts/pv/cron_chauffage.sh >> /root/scripts/pv/log_chauffage.txt 2>&1

Et voilà, le chauffage s’allume quand il faut, s’éteint quand il faut, on ne s’en occupe plus et c’est gratuit :-)

- CaSa - casa04 chez gmail point com

Mise à jour le 10 novembre 2022