Ventilation contrôlée par des framboises

Ludovic Apvrille
Axelle Apvrille

La ventilation d'une maison est indispensable pour son assainissement. L'article propose un moyen à bas coût pour ventiler automatiquement et efficacement une habitation, tout en laissant à l'utilisateur la possibilité de piloter manuellement sa ventilation depuis Internet.

1. Comment ventiler son logement sans ouvrir la fenêtre

Ventiler son logement de façon permanente est une obligation légale en France métropolitaine [1]. Cette ventilation a pour objectif de renouveler l'air des logements, notamment afin d'évacuer la vapeur d'eau produite par la respiration ou par la cuisine : cela permet de limiter l'humidité des logements, et donc la prolifération de moisissures.

Pour ventiler un logement, outre le fait d'ouvrir la fenêtre ;-), il existe deux techniques. La première, dite « naturelle », consiste à installer des conduits de tirage, en général au niveau du plafond. L'air chaud, par convection, aura alors tendance à s'échapper du logement par les conduits, provoquant une aspiration d'air neuf dans le logement. La deuxième technique consiste à installer une Ventilation Mécanique Contrôlée – VMC – afin de forcer l'air vicié du logement à travers des conduits d'évacuation.

Les VMC sont de trois types (du plus ancien, au plus récent) :

1. Les ventilations multi-vitesses, dont la vitesse est réglable manuellement. Elles possèdent en général deux vitesses, et sont pilotées par un bouton à 3 positions (arrêt, vitesse 1, vitesse 2)
2. Les ventilations à détection de taux d'humidité (dites « hygroréglables »). Ces ventilations sont multi-vitesses, et adaptent leur vitesse en fonction du taux d'humidité de l'air extrait. C'est à dire qu'un taux faible d'humidité dans le logement fait commuter la ventilation sur la petite vitesse, alors qu'un taux élevé fait commuter la ventilation sur une vitesse plus élevée.
3. Les ventilations double-flux, qui consistent à réchauffer l'air aspiré dans le logement en le faisant passer à travers un échangeur thermique qui fonctionne avec l'air extrait du logement. Ces ventilations sont en général hyg<roréglables.

Les ventilations double-flux sont plus chères à l'achat, plus complexes à installer, mais elles permettent de diminuer la perte de chaleur due à la ventilation en période hivernale. Elles ont un autre défaut : en été, l'air chaud extrait du logement pendant la nuit réchauffe l'air entrant, diminuant ainsi la capacité du logement à se refroidir la nuit.

Les ventilations hygroréglables quant à elle ont l'inconvénient de réguler le flux uniquement selon l'humidité de l'air extrait. Pas possible par exemple d'augmenter le flux pour refroidir son logement en été durant la nuit. Autre cas qui fonctionne mal : si votre logement est un peu humide, mais qu'il pleut dehors, la ventilation va tout de même accélérer l'extraction, et donc faire rentrer plus d'humidité dans votre logement …

Les ventilations multi-vitesses ont comme défaut d'exiger de la part du l'utilisateur un contrôle manuel. Les deux autres types étant soit trop chers (double-flux), soit pas assez flexibles (hygroréglables), nous avons mis en place un moyen d'automatiser de façon « intelligente » la ventilation d'un logement par utilisation d'une ventilation possédant deux vitesses, tout en laissant l'utilisateur guider la ventilation dans certains cas (cuisine odorante, etc.)

2. Objectifs de l'automatisation

Pourquoi automatiser une ventilation ? Principalement pour trois raisons :

1. Pour éviter de faire rentrer de l'humidité dans le logement, c'est à dire que si l'air qui entre risque d'augmenter le taux d'humidité du logement, alors, il vaut mieux ne pas ventiler.

2. Pour éviter de refroidir/réchauffer le logement à la mauvaise saison (consommation d'énergie supplémentaire), tout en conservant tout de même une ventilation minimale.
3. Pour profiter des possibilités de sur-ventilation. Cela permet de réchauffer la maison aux heures chaudes (automne et printemps surtout), d'éviter qu'elle ne se refroidisse trop en hiver (ventilation uniquement au delà d'une certaine température, avec une nombre d'heure minimal par jour), et enfin d'éviter le réchauffement en été, en ventilant surtout pendant les heures fraiches. Par exemple, en été, un mode pourrait être d'être en ventilation vitesse 1 par défaut , et ventilation vitesse 2 dès que température extérieure est inférieure à la température intérieure (sur-ventilation de refroidissement). D'ailleurs, outre le fait de refroidir la maison, cela donne une sensation de courant d'air, bénéfique pour résister à la chaleur.

Nous venons de voir que les paramètres de base pour automatiser de façon « intelligente » une VMC nécessite le recours à des relevés à la fois de température et d'humidité. Nous verrons à la fin de l'article que d'autres capteurs peuvent aussi être utilisés pour améliorer cette « intelligence ».

3. Architecture de l'automatisation

Bon, alors, vous voulez ventiler ? Voici le matériel pour contrôler votre ventilation !

3.1 Matériel minimal

- Une VMC bi-vitesse.
- Deux interrupteurs 250 Volt classiques. L'un des interrupteurs sert à basculer entre le mode piloté et l'ancien mode « manuel » (il est sage de garder ce mode en précaution!). L'autre permet de sélectionner la vitesse 1 ou 2.
- Deux relais capables de commander du 250V. Attention, nous recommandons fortement l'usage de relais opto-couplés, car lors de l'arrêt de la VMC, il peut y avoir un retour de charge important dû à la capacité inductive du moteur de la VMC. En effet, lors de l'arrêt de la VMC, le moteur va continuer à tourner, et donc jouer le rôle d'un alternateur qui va envoyer un courant vers les relais. Nous verrons plus tard comment le logiciel qui pilote les relais peut éviter en partie ce retour de charge (mais pas totalement …). Bref, nous avons grillés plusieurs relais non opto-couplés, donc … Aussi, certains relais se pilotent en 5V, d'autres en 3.3V, ou encore avec d'autres tensions. Le Raspberry Pi possède une alimentation externe en 5V, et une autre en 3.3V. Les GPIOs sont uniquement en 3.3V. Attention : référez-vous donc à la documentation ou au schéma de votre module à relais pour adapter la connexion entre vos relais et le Raspberry Pi.
- Un capteur d'humidité extérieure, un capteur de température extérieure, et un capteur de température intérieure.
- Un Raspberry Pi sur lequel sont connectés les capteurs (température, humidité) et chargé de commander les relais. Si vous désirez piloter la VMC depuis un site Internet / votre mobile, il faut aussi que votre carte ait un accès WIFI / Ethernet / Bluetooth. Vous pourriez également utiliser une carte de type Arduino, mais nous avons eu des problèmes de stabilité WIFI / Ethernet avec, c'est à dire qu'il était difficile de maintenir le système opérationnel pendant des mois sans reboot.

3.2 Notre configuration matérielle

3.2.1 Architecture générale

Selon la configuration de votre logement, il peut être compliqué de relever la température, l'hygrométrie et de piloter la VMC sur une même carte. C'est notre cas : nous avions déjà un Raspberry Pi (appelé par la suite RPi1) qui gérait notre station météo, mais il était difficile de piloter la VMC depuis celui ci (il aurait fallu tirer des fils – flemmingite aigüe). Nous avons donc placé un 2ème Raspberry Pi (appelé RPi2) proche de la VMC.

Ainsi, l'architecture générale de notre installation est représentée à la figure 1. Cette installation comporte :

- Des capteurs de température et d'humidité reliés par ondes radio à la base de notre station météo

(une Oregon Scientific WMR200). Cette base comporte elle-même un capteur d'humidité et de température intérieure. Cette base est connecté en USB au RPi1.

- Un RPi1 en charge de l' « intelligence ». Il relève les données issues de la station météo, envoie des ordres à RPi2 pour commander la VMC, et gère le site web de pilotage de la VMC.
- Un RPi2 en charge de la VMC. C'est lui qui, physiquement, coupe la VMC ou la met en vitesse 1 ou 2 via deux relais. Il reçoit ses ordres de RPi1 via HTTP. Un serveur web (très basique) tourne donc dessus. Le RPi2 est relié au RPi1 par Ethernet (mais l'on aurait bien sûr pu utiliser du WIFI, du Zigbee, etc.).
- Les deux relais de commande de la VMC. Le relais 1 sert à allumer la VMC, le relais 2 sert à commuter entre vitesse 1 et vitesse 2. Cette commande n'est active que si l'interrupteur de choix entre mode manuel et mode programmé est actif (voir la figure 2).

L'ensemble RPi2 et relais est visible à la Figure 3. La figure 4 présente le câblage entre RPi2 et les relais.

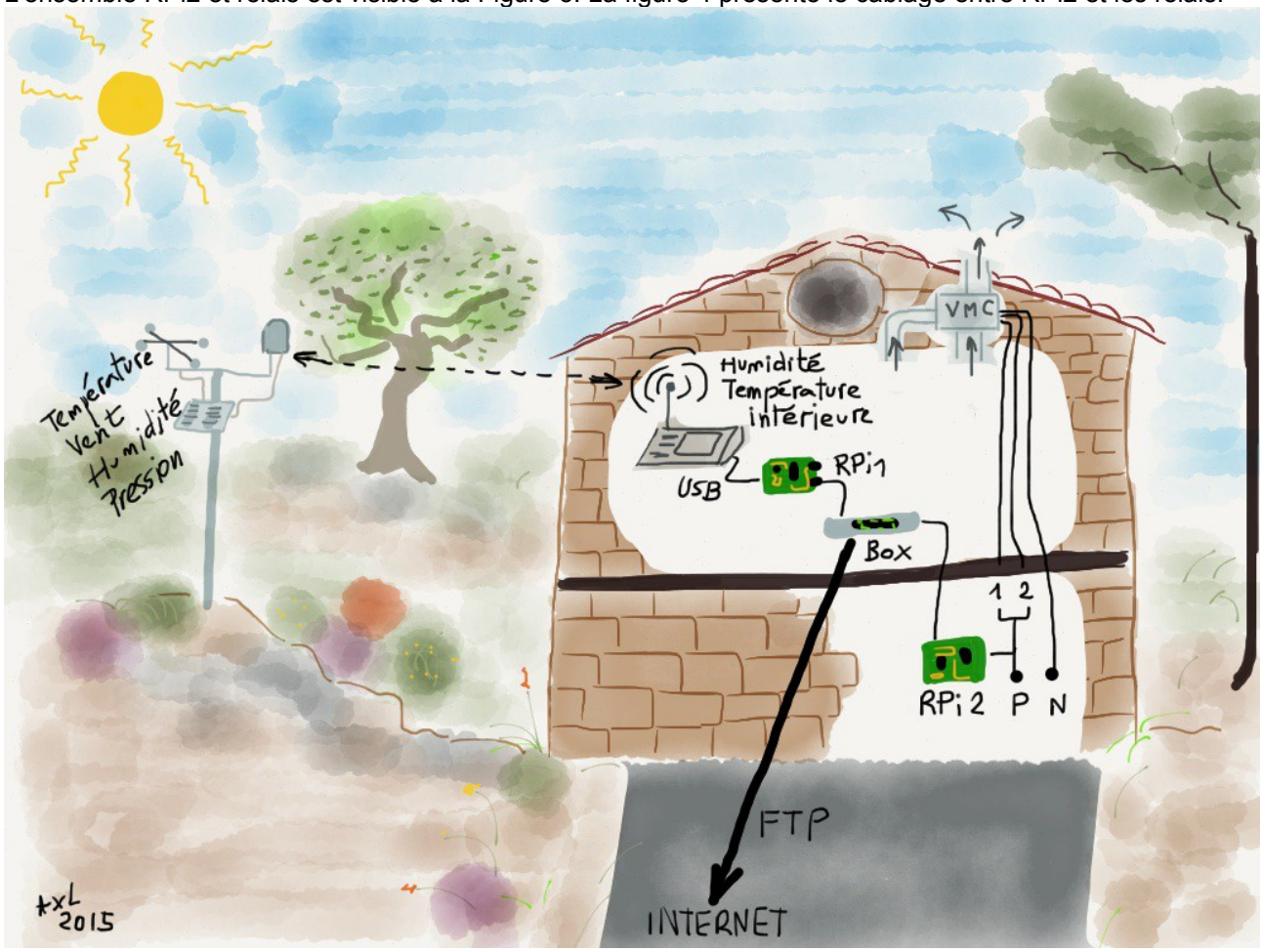

Fig. 1 : Architecture générale de l'installation.

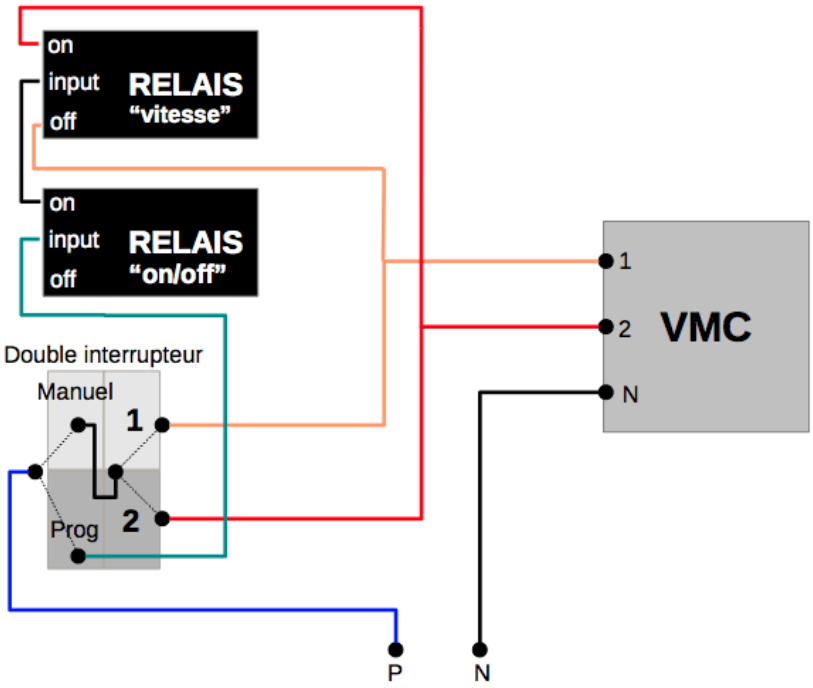

Fig. 2 : RPi2, interrupteur manuel/programme et relais

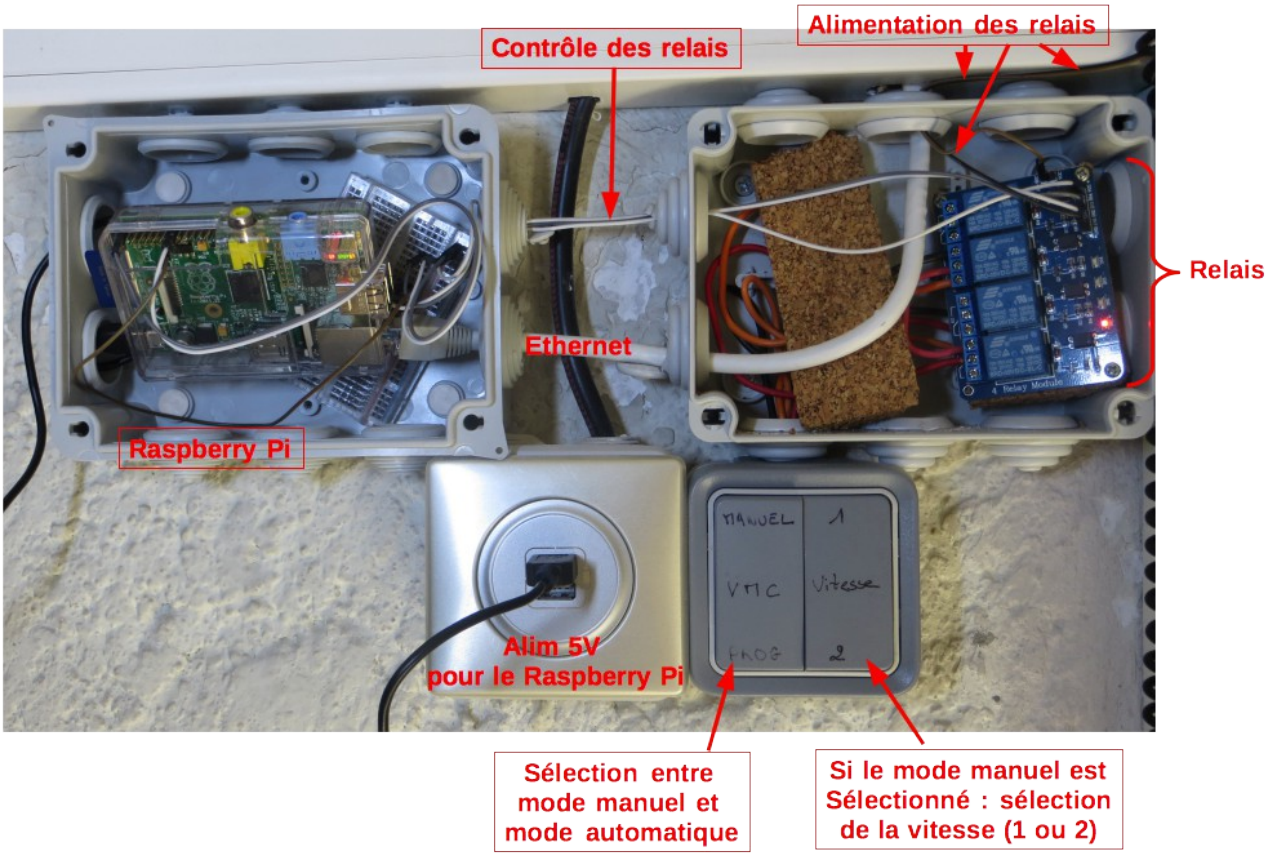

Fig. 3 : Vue de l'installation RPi2 et relais

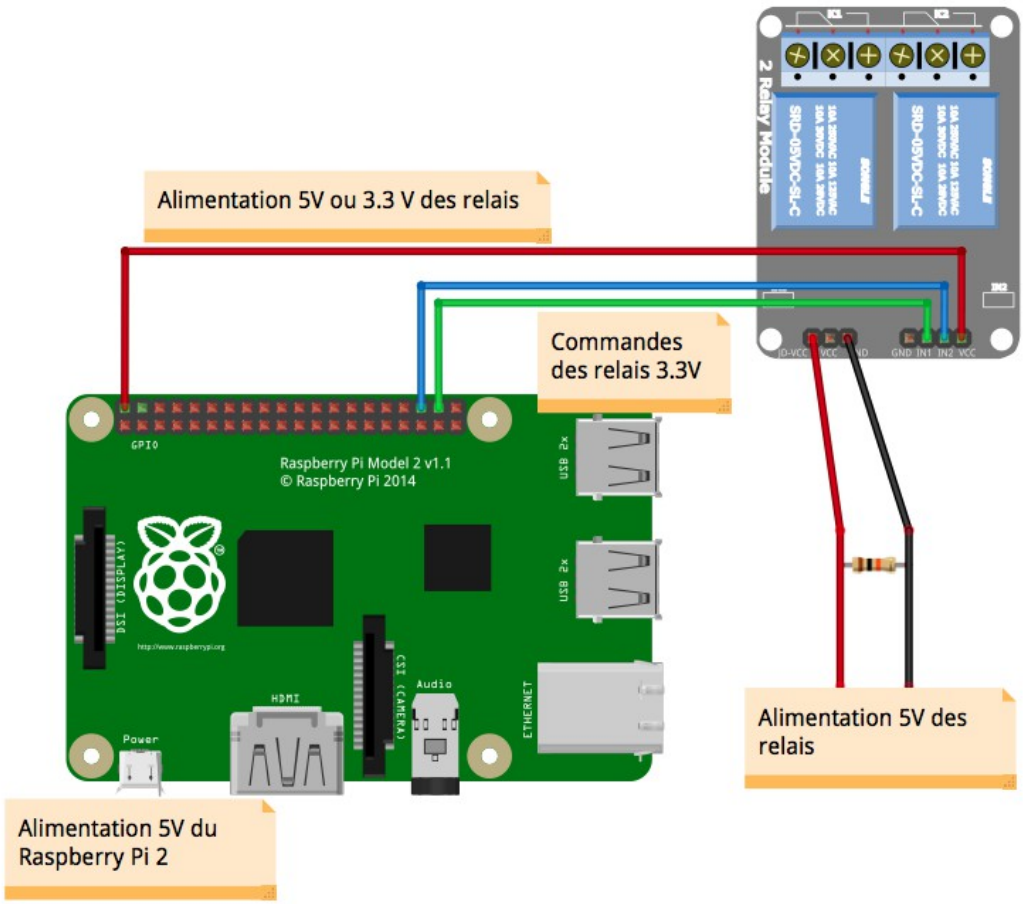

Fig. 4 : Vue de l'installation RPi2 et relais

3.2.1 Configuration logicielle du RPi2 et serveur du RPi2

Le logiciel fonctionnant sur le RPi2 est assez simple : son rôle est principalement de lire les «GET http » reçus de RPi1, d'exécuter l'ordre (activation/désactivation des relais) et de renvoyer un statut sous la forme d'une page html.

La première étape pour que le mini site Internet du RPI2 puisse commander les relais est d'activer les GPIOS du RPi2. Pour cela, en supposant que le RPi2 exécute une distribution Raspbian de Linux, il faut lancer un terminal sur le RPi2 puis configurer les GPIOs. Dans notre cas, nous utilisons les ports 22 et 23. Pour créer le port 22, il faut faire (en sudo ou root) :

```
% echo 22 > /sys/class/gpio/export
```

Il faut ensuite configurer le fait que le port 22 soit un port de sortie :

```
% echo out > /sys/class/gpio/gpio22/direction
```

Il faut, selon le groupe dans lequel tournera le site internet du RPi2, donner le droit d'accès des relais à ce groupe :

```
SUBSYSTEM=="gpio*", PROGRAM="/bin/sh -c 'chown -R root:gpio /sys/class/gpio; chmod -R 770
/sys/class/gpio; chown -R root:gpio /sys/devices/virtual/gpio; chmod -R 770
/sys/devices/virtual/gpio'"
```

Le serveur web de contrôle des relais est écrit en quelques dizaines de lignes Python. Il n'offre aucune sécurité particulière : il doit donc se trouver derrière un pare-feu, et n'autoriser que certaines adresses sources (par exemple : le RPi1).

Le code Python comporte surtout 4 parties : le démarrage du serveur web, la gestion des requêtes http, et l'activation des relais en fonction des requêtes, et les logs. L'ensemble du code est open-source, et est disponible en [2].

Commençons par le système de log : les informations du serveur web sont écrites dans les logs du

systèmes comme suit :

```
def myPrint(s):
    s1 = "[RC] %s" %(s)
    syslog.syslog(s1)
```

Le démarrage du serveur web est assez simple :

```
        myPrint('Starting relay server')
       server = HTTPServer(('', PORT), MyHandler)
        myPrint('Started httpserver...')
        server.serve_forever()
    except KeyboardInterrupt:
        myPrint('^C received, shutting down server')
        server.socket.close()
```

La gestion des requêtes consiste à extraire si le GET comporte un mot clé tel que « onRelay1 », « ofRelay1 », « onRelay2 », « ofRelay2 ». Dans le cas contraire, seul le statut est retourné. L'extrait suivant montre comment l'on peut gérer de telles requêtes :

```
class MyHandler(BaseHTTPRequestHandler):
    def do_GET(s):
         """Respond to a GET request."""
           myPrint("handling request")
         ...
           s.send_response(200)
           s.send_header("Content-type", "text/html")
           s.end_headers()
```

Les relais sont ensuite activés selon la chaine reconnue dans la requête :

```
           #Testing s
           if (s.path[1:] == "onRelay1"):
               onRelay1()
           if (s.path[1:] == "ofRelay1"):
               offRelay1()
       …
```

Enfin, la réponse est envoyée au client :

```
           myPrint("Preparing answer")
           s.wfile.write("<html><head><title>VMC SERVER</title></head>")
           req = "<p>Nb Of requests:" + str(nbOfRequests) + "</p>"
           s.wfile.write("<p>You accessed path: %s</p>" % s.path)
           s.wfile.write("<p>Uptime: %s</p>" % uptime_string)
           s.wfile.write(req)
           myPrint("Preparing relay answer")
           s.wfile.write("<p>Relay #1: %s</p>" % relay1String)
           s.wfile.write("<p>Relay #2: %s</p>" % relay2String)
           nbOfRequests +=1
           myPrint("Request: All done")
```

Voici un retour typique. Notre serveur tournait (fièrement) depuis 47 jours.

```
You accessed path: /
Uptime: 4109338.25 sec.
Nb Of requests:7081
Relay #1: ON
Relay #2: OFF
```

Le lancement automatique du serveur sur le RPi2 se fait, par exemple, en ajoutant une ligne de commande dans le script /etc/rc.local :

```
/usr/bin/python /usr/share/vmc/webserver4relays.py &
```

4. Logiciel de commandes

Pour les habitants de notre maison, il y a 6 modes différents d'utilisation de la VMC :

- Mode automatique
- OFF
- Vitesse 1 forcée
- Vitesse 2 forcée
- Pulse 1. La VMC est en vitesse 1 pendant 30 minutes, puis revient à son mode de fonctionnement précédent.

- Pulse 2. La VMC est en vitesse 2 pendant 30 minutes, puis revient à son mode de fonctionnement précédent. C'est pratique à la cuisine notamment (mais pas uniquement …) !

En mode automatique, c'est le RPi1 qui décide de l'état de la VMC. Il y a trois états possibles :

1. La VMC est arrêtée
2. La VMC est en vitesse 1
3. La VMC est en vitesse 2

Parallèlement à ces trois états possibles, nous avons défini deux politiques différentes, choisies par l'utilisateur : la politique qui consiste à conserver la chaleur, voire à réchauffer le logement – c'est le mode pour l'automne, l'hiver et le printemps -, et la politique qui consiste à rafraichir le logement (l'été). Bon, n'allez pas croire que cela fait office de chauffage ou de climatisation, mais ça aide (presque gratuitement).

4.1 Conservation de la chaleur

Conserver la chaleur consiste à ventiler pendant les heures chaudes, et à moins ventiler pendant les heures froides. Lorsque la température extérieure est supérieure à la température intérieure, on choisit alors de forcer la ventilation en vitesse 2 pour réchauffer le logement.Reste qu'en hiver, les heures les plus chaudes correspondent à un air extérieur dont la température est toujours inférieur à la température intérieure (en France métropolitaine). A contrario, en automne et au printemps, il est possible que cela se produise : l'on a alors intérêt à mettre la ventilation en vitesse 2 pour réchauffer le logement.. Le reste du temps, on définit simplement des plages horaires de fonctionnement de la VMC en vitesse 1. Enfin, il y a quelques limites : on arrête la VMC si la température extérieure est trop froide (pas forcement intéressant de ventiler par -10°C !), ou si l'humidité est trop élevée (pluie, brouillard...).

Voici notre fichier de configuration :

```
# Taux d'humidité au dessus duquel la VMC est arrêtée
maxOutHumidity=94
#Température en dessous de laquelle la VMC est arrêtée
minOutTemp=0
```

4.2 Conservation de la fraicheur

A l'inverse, la conservation de la fraicheur consiste à ventiler le logement en vitesse 2 lorsque la température extérieure est inférieure à la température intérieure. C'est donc une sur-ventilation de la maison pendant les nuits d'été. Il serait aussi bien entendu possible de définir une température au delà de laquelle on ne ventile pas le logement, mais nous ne l'avons pas fait : ainsi, dans les plages où elle est en route, la VMC ventile par défaut en vitesse 1, et en vitesse 2 pour rafraichir le logement.

4.3 Programme principal « autovmc »

Notre programme principal est écrit en Perl. Il est téléchargeable à cette adresse [2]. Nous détaillons par la suite les parties du code qui sont a priori les plus intéressantes. Nous présenterons au passage le système de log qui permet notamment de propager des informations à l'utilisateur sur le site Internet de contrôle et de configuration.

4.3.1 Vérification de la configuration

Cette section a pour objectif de vérifier que la configuration utilisateur a du sens. Par exemple, que la température minimale de ventilation n'est pas supérieure à 50 degrés.

```
sub check_config {
    my $minOutTemp = shift;
    my $maxOutHumidity = shift;

    if ($minOutTemp < -10 || $minOutTemp > 50) {
        syslog(LOG_ERR, "Incorrect value for minOutTemp=$minOutTemp");
        closelog();
        die "Incorrect value for minOutTemp";
    }
    ...
}
```

4.3.1 Relevé des valeurs météorologiques

Il s'agit là d'aller récupérer les valeurs extérieures de température et humidité, et la valeur intérieure de température. Cette récupération dépend bien entendu du type et de la configuration de votre station météo. Aussi, il faut vérifier que ces valeurs aient du sens – en cas de bug …- et que ces valeurs soient récentes, par exemple en cas d'erreur dans le relevé ou le stockage des informations … ou tout simplement si la pile d'un capteur devient trop faible. Imaginez que votre relevé de température date de 3 mois : la programmation de la VMC ne serait pas du tout adaptée !

Dans notre cas, le relevé des dernières valeurs se fait avec une requête dans le base de données SQL de stockage des valeurs météo. Ce code montre aussi le système de log mis en place. D'une part les appels à *debug_log* qui affichent les messages si le programme est démarré en mode « debug ». Et d'autre part les appels à *syslog* qui ajoutent des messages au log du système (par exemple : /var/log/messages). Cela est particulièrement important dans la dernière partie du code pour expliquer l'absence d'actions sur la VMC lorsque les relevés sont incorrects.

```
sub read_weather {
    # This retrieves the most recent entry in the database
    my $dateTime=`sqlite3 $weewx_dir/archive/weewx.sdb "select max(dateTime) from archive;"`;
    chomp($dateTime);

    # make sure dateTime is sound
    my $currentTime = time();
    print "weewxTime=$dateTime current=$currentTime\n";
    if ($currentTime - $dateTime > 10 * 60 * 60) {
        debug_log("Weewx time is obsolete");
        syslog(LOG_WARNING, "Last weewx measure too old: autovmc is quitting");
    }


    # This gets the temperature for that entry
    my $inTemp=`sqlite3 /home/weewx/archive/weewx.sdb "select inTemp from archive where dateTime=$
{dateTime};"`;

    my $outTemp = `sqlite3 /home/weewx/archive/weewx.sdb "select outTemp from archive where
dateTime=${dateTime};"`;

    my $outHumidity = `sqlite3 /home/weewx/archive/weewx.sdb "select outHumidity from archive where
dateTime=${dateTime};"`;
    chomp($inTemp);
    chomp($outTemp);
    chomp($outHumidity);

    if ($inTemp < -20 || $inTemp > 50) {
        syslog(LOG_ERR, "Inner temperature is wrong: $inTemp"); closelog();
        die "Something wrong with inner temperature: $inTemp";
    }

    if ($outTemp < -20 || $outTemp > 50) {
        syslog(LOG_ERR, "Outer temperature is wrong: $outTemp"); closelog();
        die "Something wrong with outer temperature: $outTemp";
    }

    if ($outHumidity < 0 || $outHumidity > 100) {
        syslog(LOG_ERR, "Outer humidity is wrong: $outHumidity"); closelog();
        die "Something wrong with outer humidity: $outHumidity";
    }


    debug_log("Inner temperature: $inTemp\n");
    debug_log("Outer temperature: $outTemp\n");
    debug_log("Outer humidity:    $outHumidity\n");
    return $inTemp, $outTemp, $outHumidity;
}
```

4.3.1 Fonction « warm_house »

La fonction warm_house, dont l'objectif est décrit plus haut, s'implémente de la façon suivante (pour une meilleure lisibilité, nous avons ôté les lignes qui correspondent aux logs) :

```
sub warm_house {
    my $inTemp = shift;
    my $outTemp = shift;
    my $outHumidity = shift;
    my $minOutTemp = shift;
    my $maxOutHumidity = shift;
```

```
    if (-e $pulse_indicator || (! -e $auto_indicator)) {
        syslog(LOG_NOTICE, "Manual mode or pulse detected - autovmc won't do anything");

    } else {

        if ($outTemp > $minOutTemp && $outHumidity < $maxOutHumidity) {
            if ($outTemp > $inTemp) {
                system("$vmc_dir/vmctrl.pl --high");
            } else {
                system("$vmc_dir/vmctrl.pl --low");
            }
        } else {
            system("$vmc_dir/vmctrl.pl --stop");
        }
    }

    return $inTemp, $outTemp, $outHumidity;
}
```

4.3.1 Fonction « cool_house »

La fonction « cool_house » est plus simple. Son code Perl est le suivant :

```
sub cool_house {
    my $inTemp = shift;
    my $outTemp = shift;


    if (-e $pulse_indicator || (! -e $auto_indicator)) {
        syslog(LOG_NOTICE, "Manual mode or pulse detected - autovmc won't do anything");
    } else {
        debug_log("\tAutomatic mode: let's do something smart (perhaps)\n");
        if ($outTemp < $inTemp) {
            system("$vmc_dir/vmctrl.pl --high");
        } else {
            system("$vmc_dir/vmctrl.pl --low");
        }
    }
}
```

4.4 Programmation à certaines heures

La programmation à certaines heures repose simplement sur le principe de mettre dans la crontab du RPi1 le lancement du script Python présenté précédemment. La crontab comprend en fait deux entrées :

- Une pour lancer le script « autovmc » d'exécution du mode de fonctionnement de la VMC. Dans la configuration ci-dessous, il s'exécute toute les 10 minutes de 7h du matin à 21h50.
- Une deuxième pour arrêter le VMC à 22h.

```
% crontab -l
*/10 7,8,9,10,11,12,13,14,15,16,17,18,19,20,21 * * * /home/axelle/scripts/autovmc.pl &> /dev/null
0 22 * * * /usr/share/vmc/vmctrl.pl --stop &> /dev/null
```

5. Site Internet pour le contrôle du fonctionnement et la configuration des modes

Le site Internet de la VMC a deux objectifs. Tout d'abord, piloter la VMC manuellement dans le cas où la politique automatique ne convient pas. Ensuite, visualiser l'état courant de la VMC. Cela sert d'une part à contrôler que la politique actuelle ne comporte pas de bogues, et d'autre part à vérifier que le mode en cours correspond à ce que désire l'utilisateur.

5.1 Configuration des modes de fonctionnement

Le site Internet utilisateur de la VMC fonctionne sur le RPi1. Il permet de choisir entre le mode totalement manuel (off, vitesse 1, vitesse 2, pulse1, pulse2, cancel) et le mode automatique. Un extrait de ce site est visible à la figure 5.

/// Image : SiteWebAnnoté.pdf ///

Fig. 5 : site internet de la VMC, RPI1

/// Fin légende ///

5.2 Vérification du fonctionnement de la VMC

Plusieurs informations sont disponibles pour vérifier le fonctionnement de la VMC (voir figure 6):

- Un pictogramme visuel représentant un ventilateur, qui permet d'identifier rapidement le mode de fonctionnement (OFF, 1, 2).
- Des informations de « debug » qui indiquent la valeur des relais, les températures extérieures et intérieures, et l'humidité extérieures
- Une courbe qui présente l'évolution de la vitesse de la VMC, et des paramètres météorologiques. Cela permet de vérifier les plages de fonctionnement de la VMC. Cette courbe est présentée à la figure … Cette courbe est générée avec un script python et un script *gnuplot* que nous allons à présent expliquer.

Le script Perl suivant pour objectif de récupérer les infos récentes de température, humidité et vitesse de la VMC afin de générer dans un fichier prix en entrée par le script *gnuplot*. Les principales fonctions du script Perl sont :

- Récupérer l'heure / date courante afin de vérifier que les données sont récentes

```
    my $dateTime=`sqlite3 $weewx_dir/archive/weewx.sdb "select max(dateTime) from archive;"`;
    chomp($dateTime);

    # make sure dateTime is sound
    my $currentTime = time();
    if ($currentTime - $dateTime > 10 * 60 * 60) {
        debug_log("Weewx time is obsolete");
    }
```

- Récupérer les données de température / humidité

```
# This gets the temperature for that entry
    my $inTemp=`sqlite3 /home/weewx/archive/weewx.sdb "select inTemp from archive where dateTime=$
{dateTime};"`;

    my $outTemp = `sqlite3 /home/weewx/archive/weewx.sdb "select outTemp from archive where
dateTime=${dateTime};"
`;

    my $outHumidity = `sqlite3 /home/weewx/archive/weewx.sdb "select outHumidity from archive where
dateTime=${dat
eTime};"`;
    chomp($inTemp);
    chomp($outTemp);
    chomp($outHumidity);
```

- Vérifier que ces valeurs sont valides, e.g. :

```
if ($inTemp < -20 || $inTemp > 50) {
        debug_log("Something wrong with inner temperature: $inTemp");
        $inTemp = '?';
    }
```

- Obtenir l'état de la VMC via une requête au site web de la VMC (RPi2) :

```
$res = Hijk::request({
            method       => "GET",
            host         => "$relayserver_url",
            port         => "$relayserver_port",
            path         => "$path"});
```

- Extraire de la requête http au site web du RPi2 les valeurs des relais, et donc la vitesse en cours de la VMC

```
sub read_cmv_reply {
    my $page = shift;
    debug_log("--> read_cmv_reply()\n");
    my $speed;

    if ($page eq '?') {
        return '?';
    }

    if ($page =~ /Relay #1: ON/i) {
        if ($page =~ /Relay #2: ON/i) {
            $speed = 2;
        } else {
            $speed = 1;
        }
    } else {
        $speed = 0;
    }

    debug_log("<-- read_cmv_replay(): speed=$speed\n");
    return $speed;
}
```

- Le main enchaine toutes les fonctions précédente et ajoute les valeurs à la fin d'un fichier de données

```
# --------- MAIN -----------------------------------------
my $output_file;

GetOptions ("verbose" => \$debug
     )
    or usage();

# read current weather data
my ($intemp, $outtemp, $outhumidity) = read_weather();

# read VMC speed
my $speed = get_status();

# read time
my $mytime = get_time();

print "$mytime $speed $outtemp $intemp $outhumidity\n";
```

Ce fichier de données comprend typiquement les valeurs suivantes :

```
$ more times_vmc_generated.data
11:27:05:01:01 0 3.79411764705882 17.4 43.6470588235294
...
```

selon le format Mois:jour:heure:minutes:secondes température_extérieure température_intérieure humidité_extérieure.

Le script *gnuplot* utilise deux échelles différentes : axe y1 pour la vitesse de la VMC et l'axe y2 pour les valeurs de température et d'humidité. Ensuite, il trace les courbes en fonction des valeurs du fichier de données :

```
...
set yrange [0:2.5]
set y2range [0:100]
set y2tics 10
set grid
set xlabel "Date and time"
set ylabel "Speed"
set y2label "Temperature and humidity"
set title "VMC"
set key outside right box
plot "/usr/share/vmc/curve/times_vmc_generated.data" using 1:2 index 0 title "speed" with lines ls
2, '' using 1:3
 axes x1y2  smooth bezier with lines title "Outside temp" ls 3, '' using 1:4 axes x1y2  smooth
bezier with lines t
itle "Inside temp" ls 4, '' using 1:5 axes x1y2 smooth bezier with lines title "Outside humidity" ls
5
```

Enfin, un script Bash lie le tout : il lance le script Perl de génération des valeurs, puis génère un graphe à

partir des 250 dernières valeurs générées, et enfin copie l'image générée par gnuplot au niveau du site Internet utilisateur :

```
dir=/usr/share/vmc/curve
${dir}/curve_vmc.pl >> ${dir}/times_vmc_generated.data
tail -n 250 ${dir}/times_vmc_generated.data > ${dir}/.tmp-vmc
mv ${dir}/.tmp-vmc ${dir}/times_vmc_generated.data
gnuplot ${dir}/plot_vmc.gp
mv ${dir}/vmc.png /var/www/images
```

Cette tâche est effectuée toutes les 10 minutes, via la crontab.

```
$ crontab -e
*/10 * * * * /usr/share/vmc/curve/curve_script &> /dev/null
```

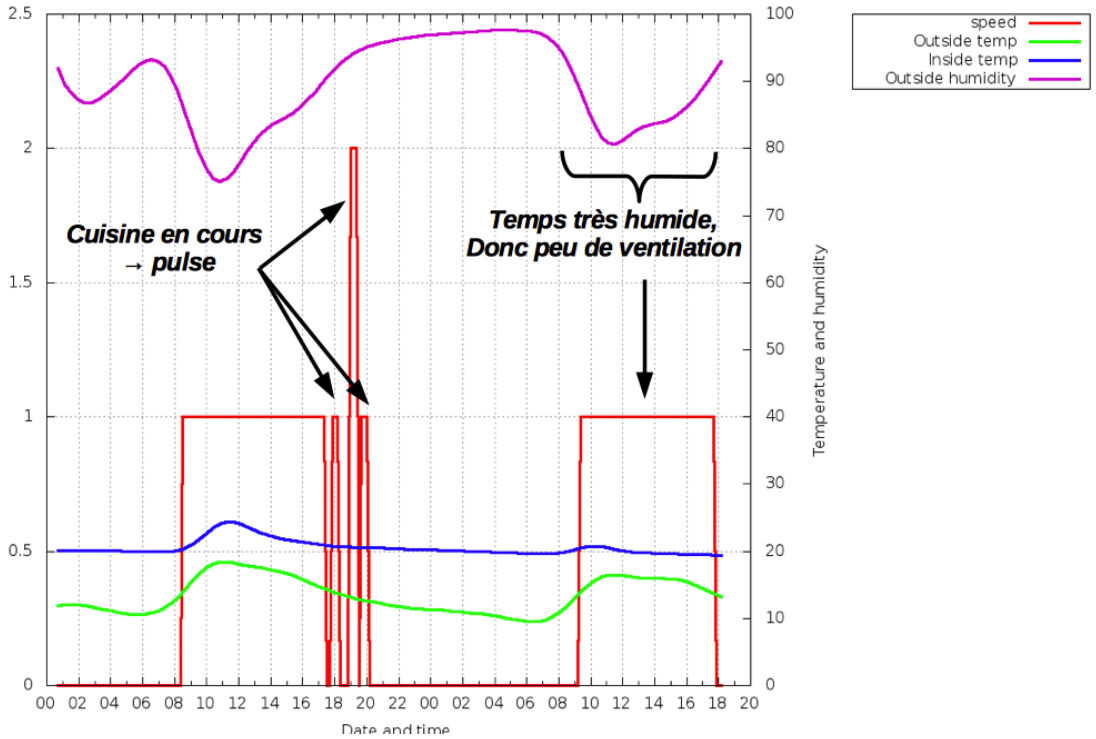

Fig. 6 : Courbe de fonctionnement de la VMC

6. Pistes d'améliorations et conclusion

L'article a montré comment éviter de faire rentrer de l'air « inadapté » (trop humide, froid, chaud) dans le logement. Néanmoins, il reste certains cas que l'automatisation ne résout pas, et qui nécessitent ainsi l'intervention de l'utilisateur, via le site Internet de la VMC, ou via les interrupteurs de commande manuelle. Voici quelques uns de ces cas, et pistes pour les résoudre.

- Evacuation des mauvaises odeurs (cuisine – lol la cuisinière est offusquée, etc.) : il faudrait bien entendu disposer de capteurs pour détecter ce problème. A défaut de disposer de tels capteurs, une idée serait sans aucun doute de disposer de capteurs de mouvement, par exemple, au niveau de la plaque de cuisson et du four, et dans les WCs. La présence d'une personne déclencherait alors la VMC tant que la personne est présente, et pour une durée pré-déterminée lorsque plus aucun mouvement n'est détecté.
- Présence d'humidité excessive après, par exemple, une douche, ou une cuisson « vapeur », ou encore après avoir passé la serpillère. La présence de nombreux invités est aussi une source d'humidité (évidemment, un pur geek n'invite personne, ne prend pas de douche et ne passe pas la serpillère ?). Là, un simple capteur d'humidité intérieur peut aider à détecter le problème – par exemple, par la montée rapide du taux d'humidité dans le logement - et à déclencher la VMC. La vitesse de la VMC peut-être adaptée en fonction du taux de montée de ce taux d'humidité.
- Présence de pollution et/ou odeur à l'extérieur du logement. L'on peut citer par exemple le cas où un

voisin fait bruler des végétaux à proximité de son logement. Dans ce cas, il faut disposer d'un capteur de pollution (taux de CO2 qui monte subitement, capteur de particules fines) qui entraine l'arrêt de la VMC. Au passage, à l'heure de la COP21, on vous signale qu'il vaudrait mieux broyer ces déchets ;)

- N'hésitez pas à nous envoyer d'autres idées !

Reste qu'en l'état actuel, nous intervenons en fait très peu sur la VMC au quotidien. La facilité d'intervention pour forcer la VMC en dehors de son mode automatique fait que nous n'avons pas l'intention pour l'instant de nous équiper d'autres capteurs. La seule amélioration possible actuellement serait de disposer d'un panneau de contrôle avec des boutons physiques comprenant une diode – un peu comme dans les centrales nucléaires ou les sous-marins -, qui permettrait de visualiser, sans son ordinateur ou son téléphone mobile, l'état de la VMC. Cela sera pour un futur numéro de hackable ;-)

[1] http://www.developpement-durable.gouv.fr/Aeration-Ventilation,12909.html

[2] https://git.framasoft.org/axellec/hackable-vmc