Kit serre connectée

Surveiller l'air, la lumière et le sol, puis arroser quand c'est nécessaire — sur un Raspberry Pi, sans nuage payant, sans abonnement.

Version 1.0 — 2026-11-18

---

1. Vue d'ensemble

Le kit mesure quatre grandeurs et commande une pompe :

- Air : température et humidité relative (SHT31, I2C)
- Lumière : éclairement en lux (BH1750, I2C)
- Sol : humidité volumétrique (sonde capacitive + ADS1115) et température (DS18B20 étanche, 1-Wire)
- Arrosage : un relais piloté par le Pi actionne une pompe ou une électrovanne

L'architecture est volontairement simple et locale :

```
capteurs ──I2C/1-Wire──> Raspberry Pi ──MQTT──> Mosquitto ──> collecteur
                                                                  |
                                                          SQLite / InfluxDB
                                                                  |
                                                           Grafana + alertes
```

Aucune donnée ne quitte le réseau local. Le Pi peut être couplé au réseau seulement pour les mises à jour.

Principe de sécurité : un actionneur ne fait rien de dangereux par défaut. Le relais est câblé ouvert = pompe arrêtée, et le logiciel ne lance un arrosage que si le sol est sec et que la cuve n'est pas vide. Un chien de garde (watchdog) coupe la pompe si le collecteur cesse d'émettre.

---

2. Matériel

#	Pièce	Référence	Prix approx.	Rôle
1	Raspberry Pi	3B+, 4 ou 5	45–90 $	cerveau
2	Capteur T°/HR	SHT31 (I2C 0x44)	8–12 $	air
3	Capteur de lumière	BH1750 (I2C 0x23)	2–4 $	lux
4	Convertisseur analogique	ADS1115 (I2C 0x48)	3–6 $	lit la sonde de sol
5	Sonde d'humidité de so	capacitive 3,3 V	4–8 $	sol
6	Sonde de température	DS18B20 étanche	3–6 $	T° sol
7	Module relais	1 canal, 5 V, optocouplé	3–6 $	pompe
8	Pompe	péristaltique 12 V ou électrovanne	15–30 $	arrosage
9	Alimentation Pi	5 V / 3 A (USB-C)	10 $	Pi
10	Alimentation pompe	12 V / 2 A	12 $	pompe
11	Résistances 4,7 kΩ, fils,	~~pl~~atine	8 $	interconnexion

Total indicatif : 70–115 $.

Deux règles qui évitent 90 % des pannes :

• La sonde de sol est capacitive (pas résistive) : elle ne se corrode pas au contact de la terre.

• La pompe a son propre alimentation ; le relais ne fait que fermer le circuit. On ne alimente jamais une pompe depuis les 5 V du Pi.

3. Câblage

3.1 Bus I2C (SHT31, BH1750, ADS1115)

```
Pi                    SHT31      BH1750     ADS1115
3V3 (pin 1) ──────────── VCC ──── VCC ──── VDD
GND (pin 6) ──────────── GND ──── GND ──── GND
SDA (GPIO2, pin 3) ──── SDA ──── SDA ──── SDA
SCL (GPIO3, pin 5) ──── SCL ──── SCL ──── SCL
```

• Les modules du commerce ont en général des résistances de *pull-up* soudées. Sinon, ajouter 4,7 kΩ de SDA vers 3V3 et 4,7 kΩ de SCL vers 3V3.

• Trois adresses I2C distinctes : SHT31 0x44, BH1750 0x23, ADS1115 0x48. Vérifié avec i2cdetect -y 1.

3.2 Sonde de sol capacitive → ADS1115

```
sonde capacitive        ADS1115
VCC (3,3 V) ──────────── (rien)
GND ──────────────────── (rien)
SIG ──────────────────── A0
```

La sonde capacitive a trois fils : VCC (2,7–5,5 V), GND, SIG. Mais attention : sur beaucoup de modules, le SIG doit être branché sur une entrée analogique — d'où l'ADS1115, le Pi n'ayant aucune entrée analogique native.

3.3 Sonde DS18B20 (1-Wire)

```
DS18B20 étanche        Pi
rouge (VDD) ──────────── 3V3 (pin 1)
noir (GND) ──────────── GND (pin 6)
jaune (DQ) ──────────── GPIO4 (pin 7)
       │
       └── 4,7 kΩ vers 3V3 (pull-up, indispensable)
```

Piège connu : sans pull-up, le bus flottant énumère des ROMs fantômes de famille 0x00 (par ex. 00-300000000000) et w1_master_attempts grimpe. Une vraie DS18B20 a une famille 0x28 et un numéro de série fixe.

3.4 Relais et pompe

```
Pi                    relais              pompe 12 V
GPIO17 (pin 11) ──────── IN
GND (pin 9) ──────────── GND
5V (pin 2) ──────────── VCC
                       COM ──────┐
                       NO ──────┼── pompe ──┐
                       NC  (non utilisé)     │
alim 12 V + ──────────────────────────────────┘
alim 12 V − ──────────────────── pompe −
```

• NO = normalement ouvert : la pompe ne tourne que quand le relais est excité. Un relais à commutation positive (active-low) est préférable : si le Pi perd l'alimentation ou se réinitialise, la pompe s'arrête.

• Ne jamais alimenter la pompe depuis le 5 V du Pi : l'appel de courant fait redémarrer le Pi (vcgencmd get_throttled ≠ 0x0).

4. Mise en route du logiciel

4.1 Activer les bus

Dans /boot/firmware/config.txt :

```
dtparam=i2c_arm=on
dtparam=spi=on
dtoverlay=w1-gpio
```

Puis redémarrer. Vérifier (ne jamais supposer) :

```
ls /dev/i2c-*                # doit montrer /dev/i2c-1
i2cdetect -y 1               # doit montrer 0x23, 0x44, 0x48
ls /sys/bus/w1/devices/      # doit montrer 28-xxxxxxxxxxxx
```

4.2 Dépendances

```
sudo apt-get update
sudo apt-get install -y mosquitto mosquitto-clients python3-smbus
sudo pip3 install --user smbus2
```

4.3 Premier test des capteurs

python3 examples/capteurs.py doit afficher des valeurs plausibles :

```
air      23,4 °C   58 %
lumière  12 430 lux
sol       41 %     18,9 °C
```

5. Les capteurs en détail

5.1 SHT31 (I2C 0x44)

Mesure unique : écrire 0x2400, attendre ~15 ms, lire 6 octets
T_MSB T_LSB T_CRC H_MSB H_LSB H_CRC.

```
T  = -45 + 175   × (T_brut / 65535)
HR = 100         × (H_brut / 65535)
```

Le CRC (polyôme 0x31, initial 0xFF) est vérifié dans le code fourni.

5.2 BH1750 (I2C 0x23)

Écrire 0x20 (mode haute résolution, 1 lx, ~120 ms), attendre, lire 2 octets :

```
lux = brut / 1,2
```

5.3 ADS1115 (I2C 0x48)

Mesure single-shot de AIN0, gain ±4,096 V, 128 SPS : registre 0x01 ← 0xC383,
puis lire 2 octets du registre 0x00 (entier signé 16 bits, 1 LSB = 125 µV).

L'humidité est une tension à étalonner :

Étape	Faire	Noter
1	sonde dans l'air sec	tension V_air
2	sonde dans un verre d'eau	tension V_eau
3	% = 100 × (V_air − V) / (V_air − V_eau)	clampé entre 0 et 100

5.4 DS18B20 (1-Wire)

```
cat /sys/bus/w1/devices/28-*/w1_slave
# ... crc=0x.. YES
# ... t=18750   → 18,75 °C
```

Si la ligne affiche NO ou crc invalide, le câblage ou la pull-up est en cause — ne pas croire la valeur.

---

6. MQTT : la colonne vertébrale

Mosquitto reçoit les mesures et les redistribue. Topics retenus :

Topic	Charge	Unité
serre/air/temperature	23,4	°C
serre/air/humidite	58	%
serre/lux	12430	lux
serre/sol/humidite	41	%
serre/sol/temperature	18,9	°C
serre/pompe	ON / OFF	—
serre/cuve	0 / 1	0 = vide

Publier à la main pour tester :

```
mosquitto_pub -t serre/air/temperature -m 23.4 -r
```

-r conserve le message : un nouveau lecteur reçoit la dernière valeur connue.

---

7. Collecteur et base de données

Le script examples/mqtt_collecteur.py s'abonne aux topics, vérifie la cohérence (température dans une plage, humidité entre 0 et 100) et écrit dans SQLite :

```
CREATE TABLE mesures (
  ts TEXT PRIMARY KEY, capteur TEXT, valeur REAL, unite TEXT
);
```

SQLite suffit largement pour une serre. InfluxDB n'est utile qu'à partir de plusieurs serres ou de plusieurs années d'historique.

Important : ne jamais écrire les valeurs brute sans vérification. Une sonde déconnectée renvoie 0 ou 65535 ; un tel valeur doit être rejetée, pas affichée.

---

8. Tableau de bord Grafana

examples/dashboard.json s'importe dans Grafana (Import → Upload JSON). Il contient quatre panneaux : température air/sol, humidité du sol, lumière et humidité de l'air.

8.1 Installer Grafana

Debian ne fournit pas Grafana : ajouter le dépôt officiel.

```
sudo install -d -m 0755 /etc/apt/keyrings
curl -sSf https://apt.grafana.com/gpg-full.key | sudo tee /etc/apt/keyrings/grafana.asc >/dev/null
echo "deb [signed-by=/etc/apt/keyrings/grafana.asc] https://apt.grafana.com stable main" \
  | sudo tee /etc/apt/sources.list.d/grafana.list
sudo apt-get update
```

Attention : l'URL du dépôt est https://apt.grafana.com stable main. Une URL erronée (avec /oss/deb) rend apt-get update en erreur ; la retirer de /etc/apt/sources.list.d/ pour revenir à la normale.

Lier Grafana à la boucle locale et interdire l'inscription :

```
sudo install -d /etc/systemd/system/grafana-server.service.d
sudo tee /etc/systemd/system/grafana-server.service.d/athena.conf >/dev/null <<'EOF'
[Service]
Environment=GF_SERVER_HTTP_ADDR=127.0.0.1
Environment=GF_USERS_ALLOW_SIGN_UP=false
Environment=GF_AUTH_ANONYMOUS_ENABLED=false
EOF
sudo systemctl daemon-reload
sudo systemctl enable --now grafana-server
```

Le mot de passe administrateur est admin par défaut : le changer ([security] admin_password ou GF_SECURITY_ADMIN_PASSWORD).

8.2 Installer le plugin SQLite

```
sudo grafana-cli --pluginsDir /var/lib/grafana/plugins \
    plugins install frser-sqlite-datasource
sudo systemctl restart grafana-server
```

8.3 Un chemin lisible par Grafana

Grafana tourne sous l'utilisateur grafana, qui ne peut pas traverser /home/<vous> (souvent en 700). Mettre la base dans un dossier partagé :

```
sudo install -d -m 2775 -o "$USER" -g grafana /var/lib/serre
cp serre.db /var/lib/serre/serre.db
sudo chown "$USER":grafana /var/lib/serre/serre.db
sudo chmod 664 /var/lib/serre/serre.db
```

8.4 Créer la source et importer

Dans Grafana : Connections → Data sources → Add → SQLite (frser-sqlite-datasource), chemin /var/lib/serre/serre.db, puis importer dashboard.json.

Deux détails du plugin, tous deux vérifiés :

- le champ de requête est queryText, pas rawSql (avec rawSql, la requête est ignorée et le panneau reste vide) ;
- pour une série temporelle, la colonne d'heure doit être numérique (epoch) et déclarée dans timeColumns, sinon Grafana ne la reconnaît pas comme temps.

C'est pour cela que les requêtes du tableau de bord utilisent strftime('%s', ts) AS time et "timeColumns": ["time"].

9. Arrosage et sécurité

9.1 Règle d'arrosage

```
SI sol < 25 % ET cuve = 1 ET heure dans [06:00, 21:00]
ALORS pompe ON pendant 30 s
SINON pompe OFF
```

- Anti-marie-sèche : la pompe péristaltique ne doit jamais tourner à sec. Un contact de niveau dans la cuve (ou une sonde à deux fils) coupe le relais.

• Anti-inondation : un arrosage est limité dans le temps, et un second cycle exige que le sol ait eu le temps d'absorber (délai minimal 10 min).
• Chien de garde : si le collecteur n'a pas émis depuis 10 min, un service systemd coupe la pompe.

9.2 Le relais

Le relais est le seul composant qui manipule de la puissance. Trois règles :

1. ne jamais le piloter depuis une sortie 5 V logique si la pompe est en 12 V sans alimentation séparée ;
2. ne jamais utiliser la sortie NC (normalement fermée) pour la pompe ;
3. mettre une résistance de 220 Ω entre GPIO17 et IN si le module n'a pas de optocoupleur.

---

10. Mise en service pas à pas

1. Monter le matériel, tout débranché.
2. Activer I2C/1-Wire, redémarrer, vérifier i2cdetect et /sys/bus/w1.
3. Lancer capteurs.py : valeurs plausibles ?
4. Démarrer Mosquitto, publier un message de test.
5. Lancer le collecteur, vérifier la table SQLite.
6. Importer le tableau de bord.
7. Enfin seulement, brancher le relais et la pompe, et tester l'arrosage.

---

11. Dépannage

Symptôme	Cause probable	Vérification
i2cdetect ne montre rien	bus non activé, pull-up absente, câble	ls /dev/i2c-1, mesurer 3,3 V sur SDA au repos
ROM 00-… sur 1-Wire	bus flottant	pull-up 4,7 kΩ, cat …/w1_slave doit dire YES
0 ou 65535 en permanence	sonde débranchée	re-vérifier le CRC
humidité de sol aberrante	étalonnage absent	refaire l'étalonnage air/eau
pompe qui redémarre le Pi	alim commune	alim 12 V séparée
get_throttled ≠ 0x0	alim insuffisante	vcgencmd measure_volts

---

12. Annexes

Adresses I2C

Capteur	Adresse	Broche ADDR
SHT31	0x44	GND
SHT31	0x45	VDD
BH1750	0x23	GND
BH1750	0x5C	VDD
ADS1115	0x48	GND
ADS1115	0x49	VDD
ADS1115	0x4A	SDA
ADS1115	0x4B	SCL

Brochage utilisé

Fonction	BCM	Broche physique
SDA (I2C)	GPIO2	3
SCL (I2C)	GPIO3	5
1-Wire	GPIO4	7
Pompe (relais)	GPIO17	11
3V3	—	1
5V	—	2
GND	—	6, 9

Limites connues

- Le kit est conçu pour une serre d'appartement ou de jardin, pas pour une production commerciale.
- La sonde capacitive vieillit : remplacer tous les 2–3 ans.
- Aucune garantie sur les prix indiqués, fournis à titre indicatif.