Contents

Filament Storage Environmental Manager (FSEM) Installation Guide

Complete installation and configuration guide for FSEM.

Table of Contents

System Requirements

Operating System

- **Primary Target**: Raspberry Pi OS (Debian-based)
- **Supported**: Ubuntu, Debian, RedHat/CentOS, Fedora
- **Architecture**: ARM (Raspberry Pi) or x86_64

Software Requirements

- **Python**: 3.13 (recommended) or 3.11+
- **SQLCipher**: For encrypted configuration (pysqlcipher3)
- **Database Backend**: One of:
  - InfluxDB v1/v2/v3 (can be remote)
  - Prometheus (push gateway)
  - TimescaleDB (PostgreSQL-based)
  - VictoriaMetrics
  - None (local-only mode)
- **systemd**: For service management

- **nginx**: Optional, for web UI reverse proxy
- **HashiCorp Vault**: Optional, for enterprise secret management (hvac library)

Network Requirements

- Network access to database backend (if using remote database)
- Optional: HashiCorp Vault server access
- Optional: Web UI access (port 5000 or custom nginx config)

Hardware Requirements

Required Hardware

- Raspberry Pi (Zero W, 3, 4, or 5) or compatible SBC
- One of the following sensors:
  - **BME280** - I2C temperature/humidity/pressure sensor
  - **DHT22** - GPIO-based temperature/humidity sensor
- MicroSD card (8GB+ recommended)
- 5V power supply

Optional Hardware

For environmental control features: - **Heating Relay Module** (5V or 3.3V compatible) - Connected to GPIO pin 16 (configurable) - For temperature control - **Fan Relay Module** (5V or 3.3V compatible) - Connected to GPIO pin 20 (configurable) - For humidity control

Sensor Connections

BME280 (I2C)

```
BME280 Pin      Raspberry Pi Pin
----------------------------------
VCC/VIN      -> 3.3V (Pin 1 or 17)
GND          -> Ground (Pin 6, 9, 14, 20, 25, 30, 34, or 39)
SDA          -> GPIO 2 (SDA, Pin 3)
SCL          -> GPIO 3 (SCL, Pin 5)
```

DHT22 (GPIO)

```
DHT22 Pin       Raspberry Pi Pin
----------------------------------
VCC/+        -> 3.3V or 5V (Pin 1, 2, 4, or 17)
DATA         -> GPIO 4 (Pin 7) - configurable
GND          -> Ground (Pin 6, 9, 14, 20, 25, 30, 34, or 39)
```

Note: A 10kohm pull-up resistor between DATA and VCC is recommended for DHT22.

Relay Modules (Optional)

```
Heating Relay (Default GPIO 16, Pin 36)
Fan Relay (Default GPIO 20, Pin 38)

Relay Pin       Raspberry Pi Pin
----------------------------------
VCC          -> 5V (Pin 2 or 4)
GND          -> Ground (any GND pin)
```

```
IN/Signal    -> GPIO 16 or 20 (configurable)
```

Safety Warning: - Ensure relay modules are properly rated for your heating/fan devices - Use appropriate electrical isolation - Follow all local electrical codes - Never work on live electrical circuits - Consider using an electrician for mains voltage installations

---

Quick Installation

The master installer (`install.sh`) handles everything: directory setup, virtual environment, dependencies, configuration, and service installation.

Step 1: Download the Repository

```
# Clone the repository
git clone https://github.com/jdelgado-dtlabs/filamentenvmonitor.git
cd filamentenvmonitor
```

Or download and extract the latest release:

```
wget https://github.com/jdelgado-dtlabs/filamentenvmonitor/archive/refs/tags/v2.0.0.tar.gz
tar -xzf v2.0.0.tar.gz
cd filamentenvmonitor-2.0.0
```

Step 2: Run the Master Installer

```
sudo ./install/install.sh
```

Step 3: Follow Interactive Prompts

The installer will guide you through:

1. **Installation Directory Selection**:
    - Default: `/opt/filamentcontrol`
    - Current directory
    - Custom path
2. **Configuration Setup** (automated via `setup.sh`):
    - Vault configuration (optional)
    - Encryption key generation
    - Database backend selection (7 options)
    - Sensor type and pins
    - Optional heating/humidity control
3. **Virtual Environment & Dependencies**:
    - Creates Python venv
    - Installs all dependencies
    - Installs SQLCipher and Vault libraries
4. **Service Installation**:
    - Generates service files with dynamic paths
    - Installs main and web UI services
    - Optionally starts services

Step 4: Reconfiguration (Optional)

To modify configuration after installation:

```
cd /opt/filamentcontrol
sudo ./install/setup.sh
```

Configuration Options: 1. **Reconfigure everything** - Regenerate encryption keys, Vault, database, sensor settings 2. **Modify specific settings** - Interactive menu for individual changes 3. **Exit** - Keep current configuration

The installer will ask:

1. **Installation Directory**:

   ```
   Select installation directory:
   1) /opt/filamentcontrol (default)
   2) /opt/filamentcontrol (current location)
   3) Enter custom path
   Enter choice [1]:
   ```

2. **Service Start Preference**:

   ```
   Do you want to start the filamentbox service now? [Y/n]:
   Do you want to start the web UI service now? [Y/n]:
   ```

What the Installer Does

1. **Directory Setup**:
   - Creates installation directory if it doesn't exist
   - Copies all application files
   - Sets proper ownership and permissions
2. **Encryption & Security**:
   - Generates 64-character encryption key (384 bits entropy)
   - Stores key in `.config_key` with 600 permissions
   - Optionally stores key in HashiCorp Vault
   - Creates encrypted SQLCipher configuration database
3. **Configuration**:
   - Interactive database backend selection
   - Sensor type and connection configuration
   - Optional heating/humidity control setup
   - Automatic migration from legacy YAML/.env files
4. **Service Generation**:
   - Auto-generates systemd service files with dynamic paths
   - Embeds Vault environment variables if configured
   - Configures Python virtual environment path
   - Sets working directory based on installation location
5. **Virtual Environment**:
   - Checks for existing virtual environment
   - Creates new environment if needed
   - Installs Python dependencies (including pysqlcipher3)
   - Installs hvac library if Vault configured
6. **Service Installation**:
   - Installs main application service
   - Installs web UI service
   - Enables services for auto-start
7. **Verification**:
   - Checks service status
   - Shows logs if any issues occur
   - Provides next steps and key backup reminder

Post-Installation

After successful installation:

```
# Check service status (Web UI is integrated in main service)
sudo systemctl status filamentbox.service

# View logs
sudo journalctl -u filamentbox.service -f

# Access web UI
# Open browser to: http://YOUR_PI_IP:5000
```

Manual Installation

For users who prefer manual control or custom setups.

Step 1: System Preparation

```
# Update system packages
sudo apt update
sudo apt upgrade -y

# Install system dependencies (Debian/Ubuntu)
sudo apt install -y python3.13 python3.13-venv python3-pip git i2c-tools \
    libsqlcipher-dev sqlcipher

# For RedHat/CentOS
sudo yum install -y python3.13 python3-pip git i2c-tools sqlcipher sqlcipher-devel
```

Note: SQLCipher libraries are required for encrypted configuration database support.

Step 2: Enable I2C (for BME280)

```
# Edit boot config
sudo nano /boot/config.txt

# Add or uncomment this line:
dtparam=i2c_arm=on

# Reboot to apply changes
sudo reboot

# Verify I2C is enabled
ls /dev/i2c-*
# Should show: /dev/i2c-1

# Scan for I2C devices (BME280 usually at 0x76 or 0x77)
i2cdetect -y 1
```

Step 3: Create Installation Directory

```
# Create directory structure
sudo mkdir -p /opt/filamentcontrol
sudo chown $USER:$USER /opt/filamentcontrol
cd /opt/filamentcontrol
```

```
# Clone repository
git clone https://github.com/jdelgado-dtlabs/filamentenvmonitor.git .
```

Step 4: Set Up Python Environment

```
# Create virtual environment
python3.13 -m venv filamentcontrol

# Activate virtual environment
source filamentcontrol/bin/activate

# Install runtime dependencies
pip install -r requirements.txt

# Optional: Install development tools
pip install -r requirements-dev.txt
pre-commit install
```

Step 5: Configure Application

```
# Run setup script for guided configuration
sudo ../install/setup.sh
```

See Configuration Guide below for detailed configuration options.

Step 6: Test Installation

```
# Test sensor reading
python -m filamentbox.main --debug

# Should see output like:
# INFO - Starting filament environment monitor...
# INFO - Sensor initialized: BME280
# DEBUG - Temperature: 20.5C, Humidity: 45.2%, Pressure: 1013.25 hPa
```

Press Ctrl+C to stop.

Step 7: Install as Service

```
# Run service installer
sudo ./install/install_service.sh

# Run web UI installer
sudo ./install/install_webui_service.sh
```

Configuration Guide

v2.0 Encrypted Configuration

All configuration is now stored in an **encrypted SQLCipher database** (`config.db`) instead of plain-text YAML files. This provides: - 256-bit AES encryption for all sensitive data - Type-safe value storage with automatic inference - No credentials in version control - Optional HashiCorp Vault integration

Configuration Tool

Use `setup.sh` to manage all configuration settings:

```
cd /opt/filamentcontrol
sudo ./install/setup.sh
```

Configuration Management: - **Full Reconfiguration**: Regenerate encryption keys, Vault setup, database, sensor - **Modify Settings**: Interactive menu for specific configuration changes - **Service Regeneration**: Auto-generates service files with updated paths/Vault vars - `N` - **Next** section (cycle through all sections) - `S` - **Search** for a key by name - `V` - **View** all configuration - `E` - **Edit** current section values - `D` - **Delete** a configuration key - `C` - **Create** a new configuration key - `Q` - **Quit**

Command-Line Configuration:

```
# View all configuration
python scripts/config_tool.py --list

# Get specific value
python scripts/config_tool.py --get database.type
python scripts/config_tool.py --get sensor.type

# Set value (type automatically inferred)
python scripts/config_tool.py --set database.influxdb_v2.org myorg
python scripts/config_tool.py --set sensor.type BME280
python scripts/config_tool.py --set heating_control.enabled true
python scripts/config_tool.py --set heating_control.min_temp_c 18.0

# Delete value
python scripts/config_tool.py --delete unwanted.key
```

Database Backend Configuration

Choose from 7 database backends. Configure during setup or change anytime with the config tool.

InfluxDB v1 (Legacy)

```
# Configure via setup.sh
sudo ./install/setup.sh
# Select: InfluxDB v1
# Enter: host, port, database, username, password
```

InfluxDB v2 (Modern)

```
python scripts/config_tool.py --set database.type influxdb_v2
python scripts/config_tool.py --set database.influxdb_v2.url "http://192.168.1.10:8086"
python scripts/config_tool.py --set database.influxdb_v2.org "myorg"
python scripts/config_tool.py --set database.influxdb_v2.bucket "filamentbox"
python scripts/config_tool.py --set database.influxdb_v2.token "your-influxdb-token"
```

InfluxDB v3 (Latest)

```
python scripts/config_tool.py --set database.type influxdb_v3
python scripts/config_tool.py --set database.influxdb_v3.host "us-east-1-1.aws.cloud2.influxdata.com"
python scripts/config_tool.py --set database.influxdb_v3.database "filamentbox"
python scripts/config_tool.py --set database.influxdb_v3.token "your-v3-token"
```

Prometheus (Push Gateway)

```
# Configure via setup.sh
sudo ./install/setup.sh
# Select: Prometheus
# Enter: push gateway URL, job name
```

TimescaleDB (PostgreSQL)

```
python scripts/config_tool.py --set database.type timescaledb
python scripts/config_tool.py --set database.timescaledb.host "192.168.1.10"
python scripts/config_tool.py --set database.timescaledb.port 5432
python scripts/config_tool.py --set database.timescaledb.database "filamentbox"
python scripts/config_tool.py --set database.timescaledb.user "postgres"
python scripts/config_tool.py --set database.timescaledb.password "secret"
python scripts/config_tool.py --set database.timescaledb.table "environment"
```

VictoriaMetrics

```
python scripts/config_tool.py --set database.type victoriametrics
python scripts/config_tool.py --set database.victoriametrics.url "http://192.168.1.10:8428"
```

None (Local-Only Mode)

```
python scripts/config_tool.py --set database.type none
# No remote database - data logged locally only
```

Sensor Configuration

BME280 (I2C):

```
python scripts/config_tool.py --set sensor.type BME280
python scripts/config_tool.py --set sensor.sea_level_pressure 1013.25
```

DHT22 (GPIO):

```
python scripts/config_tool.py --set sensor.type DHT22
python scripts/config_tool.py --set sensor.gpio_pin 4
```

DHT11 (GPIO):

```
python scripts/config_tool.py --set sensor.type DHT11
python scripts/config_tool.py --set sensor.gpio_pin 4
```

Data Collection Configuration

```
python scripts/config_tool.py --set data_collection.read_interval 5
python scripts/config_tool.py --set data_collection.batch_size 10
python scripts/config_tool.py --set data_collection.flush_interval 60
python scripts/config_tool.py --set data_collection.measurement "environment"

# Tags (use interactive editor for key-value pairs)
python scripts/config_tool.py --interactive
# Navigate to data_collection section, Edit tags with special tag editor
```

Temperature & Humidity Control

Heating Control:

```
python scripts/config_tool.py --set heating_control.enabled true
python scripts/config_tool.py --set heating_control.gpio_pin 16
python scripts/config_tool.py --set heating_control.min_temp_c 18.0
python scripts/config_tool.py --set heating_control.max_temp_c 22.0
python scripts/config_tool.py --set heating_control.check_interval 10
```

Humidity Control:

```
python scripts/config_tool.py --set humidity_control.enabled true
python scripts/config_tool.py --set humidity_control.gpio_pin 20
python scripts/config_tool.py --set humidity_control.min_humidity 40.0
python scripts/config_tool.py --set humidity_control.max_humidity 60.0
python scripts/config_tool.py --set humidity_control.check_interval 10
```

Hysteresis Behavior: - Heater turns ON when temp < `min_temp_c` - Heater turns OFF when temp > `max_temp_c` - Between min/max: maintains current state (prevents rapid cycling)

Queue & Retry Configuration

```
python scripts/config_tool.py --set queue.max_size 1000
python scripts/config_tool.py --set retry.backoff_base 2.0
python scripts/config_tool.py --set retry.backoff_max 300.0
python scripts/config_tool.py --set retry.alert_threshold 5
python scripts/config_tool.py --set retry.persist_on_alert true
```

Persistence Configuration

```
python scripts/config_tool.py --set persistence.db_path "unsent_batches.db"
python scripts/config_tool.py --set persistence.max_batches 100
```

Encryption Key Management

The encryption key is automatically generated and stored during setup. For details on key storage, loading priority, and recovery, see:

- Encryption Key Security Guide
- HashiCorp Vault Integration Guide

Key Loading Priority: 1. `CONFIG_ENCRYPTION_KEY` environment variable 2. HashiCorp Vault (if configured) 3. `.config_key` file (local storage) 4. Default key (insecure, not recommended)

Vault Configuration:

```
# Set Vault environment variables
export VAULT_ADDR="https://vault.example.com:8200"
export VAULT_TOKEN="your-vault-token"

# Or configure in service files (auto-generated by setup.sh)
```

Legacy Configuration Migration

If you have existing `config.yaml` or `.env` files from v1.x:

```
# Run migration script
python scripts/migrate_config.py

# Or run setup.sh which migrates automatically
sudo ./install/setup.sh
```

Migration process: 1. Reads all values from YAML and .env files 2. Imports into encrypted database 3. Backs up legacy files with timestamp 4. Removes legacy files 5. All settings preserved

Hysteresis Example: - Fan turns ON when humidity > 60.0% - Fan turns OFF when humidity < 40.0% - Between 40.0-60.0%: maintains current state

Complete Configuration Example

```yaml
influxdb:
  host: "192.168.1.10"
  port: 8086
  database: "filamentbox"
  username: "admin"
  password: "changeme"

data_collection:
  read_interval: 5
  batch_size: 10
  flush_interval: 60
  measurement: "environment"
  tags:
    location: "filamentbox"
    device: "pi-zero-w"
    room: "workshop"

sensor:
  type: "bme280"
  sea_level_pressure: 1013.25

queue:
  max_size: 1000

retry:
  backoff_base: 2.0
  backoff_max: 300.0
  alert_threshold: 5
  persist_on_alert: true

persistence:
  db_path: "unsent_batches.db"
  max_batches: 100

heating_control:
  enabled: true
  gpio_pin: 16
  min_temp_c: 18.0
  max_temp_c: 22.0
  check_interval: 10

humidity_control:
  enabled: true
  gpio_pin: 20
  min_humidity: 40.0
  max_humidity: 60.0
  check_interval: 10
```

Production Deployment

Service Installation

Both service installers support version detection and smart updates.

Main Application Service

```
# Run the service installer
sudo ./install/install_service.sh
```

What it does: 1. Detects operating system (Debian/Ubuntu or RedHat/CentOS) 2. Installs required system packages: - Debian/Ubuntu: `python3-dev`, `python3-venv`, `i2c-tools` - RedHat/CentOS: `python3-devel`, `i2c-tools` 3. Checks for existing service installation 4. Compares versions (if service exists) 5. Shows diff of changes (if updating) 6. Copies service file to `/etc/systemd/system/` 7. Reloads systemd daemon 8. Enables service for auto-start 9. Optionally starts service immediately

Service file location: `/etc/systemd/system/filamentbox.service`

Note: The Web UI is integrated into the main service via the orchestrator. There is no separate webui service in v2.0+.

Nginx configuration (optional for reverse proxy): - Docker: `/etc/nginx/conf.d/filamentbox.conf` - Bare metal: `/etc/nginx/sites-available/filamentbox` (symlinked to sites-enabled)

Nginx Configuration

Automatic Configuration The web UI installer automatically configures nginx if detected:

```
# Docker installation
# /etc/nginx/conf.d/filamentbox.conf

server {
    listen 80;
    server_name _;

    location / {
        proxy_pass http://127.0.0.1:5000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}

# Bare metal installation
# /etc/nginx/sites-available/filamentbox

server {
    listen 80;
    server_name _;

    location / {
        proxy_pass http://127.0.0.1:5000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

Manual Nginx Configuration If you prefer manual configuration or need HTTPS:

```
# Create configuration file
sudo nano /etc/nginx/sites-available/filamentbox

# Add configuration (see above)

# Enable site (bare metal only)
sudo ln -s /etc/nginx/sites-available/filamentbox /etc/nginx/sites-enabled/

# Test configuration
sudo nginx -t

# Reload nginx
sudo systemctl reload nginx
```

HTTPS Configuration For HTTPS with Let's Encrypt:

```
# Install certbot
sudo apt install certbot python3-certbot-nginx

# Obtain certificate (replace your-domain.com)
sudo certbot --nginx -d your-domain.com

# Certbot will automatically update nginx configuration
# and set up auto-renewal
```

Manual HTTPS configuration:

```
server {
    listen 443 ssl http2;
    server_name your-domain.com;

    ssl_certificate /etc/ssl/certs/your-cert.crt;
    ssl_certificate_key /etc/ssl/private/your-key.key;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers HIGH:!aNULL:!MD5;

    location / {
        proxy_pass http://127.0.0.1:5000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}

server {
    listen 80;
    server_name your-domain.com;
    return 301 https://$server_name$request_uri;
}
```

Firewall Configuration

```bash
# UFW (Ubuntu/Debian)
sudo ufw allow 5000/tcp        # Direct Flask access
sudo ufw allow 80/tcp          # HTTP (nginx)
sudo ufw allow 443/tcp         # HTTPS (nginx)
sudo ufw enable

# firewalld (RedHat/CentOS)
sudo firewall-cmd --permanent --add-service=http
sudo firewall-cmd --permanent --add-service=https
sudo firewall-cmd --permanent --add-port=5000/tcp
sudo firewall-cmd --reload
```

Service Management

Systemd Commands

Main Application Service

```bash
# Start service
sudo systemctl start filamentbox.service

# Stop service
sudo systemctl stop filamentbox.service

# Restart service
sudo systemctl restart filamentbox.service

# Check status
sudo systemctl status filamentbox.service

# Enable auto-start on boot
sudo systemctl enable filamentbox.service

# Disable auto-start
sudo systemctl disable filamentbox.service

# View logs (real-time)
sudo journalctl -u filamentbox.service -f

# View logs (last 100 lines)
sudo journalctl -u filamentbox.service -n 100

# View logs (since boot)
sudo journalctl -u filamentbox.service -b
```

Web UI Service

```bash
# Start service
sudo systemctl start filamentbox.service (v2.0+ - webui integrated)
```

```
# Stop service
sudo systemctl stop filamentbox.service (v2.0+ - webui integrated)

# Restart service
sudo systemctl restart filamentbox.service (v2.0+ - webui integrated)

# Check status
sudo systemctl status filamentbox.service (v2.0+ - webui integrated)

# Enable auto-start on boot
sudo systemctl enable filamentbox.service (v2.0+ - webui integrated)

# Disable auto-start
sudo systemctl disable filamentbox.service (v2.0+ - webui integrated)

# View logs (real-time)
sudo journalctl -u filamentbox.service (v2.0+ - webui integrated) -f

# View logs (last 100 lines)
sudo journalctl -u filamentbox.service (v2.0+ - webui integrated) -n 100
```

Service File Locations

```
/etc/systemd/system/filamentbox.service
/etc/systemd/system/filamentbox.service (v2.0+ - webui integrated)
```

Service File Details

Main Service (filamentbox.service)

```
# Version: 2.0.0
[Unit]
Description=Filament Storage Environmental Manager (with integrated Web UI)
After=network.target

[Service]
Type=simple
User=root
WorkingDirectory=/opt/filamentcontrol
Environment="PATH=/opt/filamentcontrol/filamentcontrol/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/us
ExecStart=/opt/filamentcontrol/filamentcontrol/bin/python -m filamentbox.main
Restart=on-failure
RestartSec=10
StandardOutput=journal
StandardError=journal

# Security hardening
ProtectSystem=strict
ReadWritePaths=/opt/filamentcontrol
PrivateTmp=true

[Install]
WantedBy=multi-user.target
```

Note: The Web UI server is integrated into the main service via the orchestrator in v2.0+. There is no separate webui service file.

Security hardening

ProtectSystem=strict ReadWritePaths=/opt/filamentcontrol PrivateTmp=true

[Install] WantedBy=multi-user.target

~~~~
**Key Features**:
- **Version tracking**: Each service file includes version number
- **Auto-restart**: Services restart automatically on failure
- **Security hardening**: Restricted file system access, isolated temp files
- **Proper logging**: Outputs to systemd journal
- **Service dependency**: Web UI requires main service to be running

---

## Updating Existing Installation

Both installers include smart version detection and graceful updates.

### Automatic Update Process

```bash
# Pull latest changes
cd /opt/filamentcontrol
git pull origin master

# Update main service
sudo ./install/install_service.sh

# Update web UI service
sudo ./install/install_webui_service.sh
~~~~

What Happens During Update

1. **Version Detection**:

   ```
   Existing service detected: /etc/systemd/system/filamentbox.service
   Current version: 1.5.0
   New version: 1.6.0
   ```

2. **Change Preview**:

   ```
   Changes in new version:
   - Updated Python path
   - Added security hardening flags
   - Modified restart policy
   ```

3. **Service State Preservation**:

   - If service was running: gracefully stops, updates, restarts
   - If service was stopped: updates without starting

4. **Verification**:

```
Service updated successfully!
Status: active (running)
```

Manual Service Update

If you prefer manual control:

```
# Stop services
sudo systemctl stop filamentbox.service (v2.0+ - webui integrated)
sudo systemctl stop filamentbox.service

# Update code
git pull origin master

# Update dependencies
source filamentcontrol/bin/activate
pip install -r requirements.txt

# Copy new service files
sudo cp filamentbox.service /etc/systemd/system/
sudo cp filamentbox.service (v2.0+ - webui integrated) /etc/systemd/system/

# Reload systemd
sudo systemctl daemon-reload

# Start services
sudo systemctl start filamentbox.service
sudo systemctl start filamentbox.service (v2.0+ - webui integrated)

# Verify status
sudo systemctl status filamentbox.service
sudo systemctl status filamentbox.service (v2.0+ - webui integrated)
```

Rolling Back Updates

If an update causes issues:

```
# Stop services
sudo systemctl stop filamentbox.service (v2.0+ - webui integrated)
sudo systemctl stop filamentbox.service

# Roll back code
cd /opt/filamentcontrol
git log --oneline  # Find previous version commit hash
git checkout <previous-commit-hash>

# Reinstall services
sudo ./install/install_service.sh
sudo ./install/install_webui_service.sh

# Or manually restore old service files if you have backups
sudo cp /path/to/backup/filamentbox.service /etc/systemd/system/
sudo systemctl daemon-reload
sudo systemctl start filamentbox.service
```

Troubleshooting

Common Installation Issues

Python Version Not Found **Problem**: `python3.13: command not found`

Solution:

```
# Add deadsnakes PPA (Ubuntu/Debian)
sudo add-apt-repository ppa:deadsnakes/ppa
sudo apt update
sudo apt install python3.13 python3.13-venv

# Or use available Python version (3.11+)
python3.11 -m venv filamentcontrol
```

I2C Not Enabled **Problem**: `FileNotFoundError: [Errno 2] No such file or directory: '/dev/i2c-1'`

Solution:

```
# Enable I2C
sudo raspi-config
# Navigate to: Interface Options > I2C > Enable

# Or manually edit config
sudo nano /boot/config.txt
# Add: dtparam=i2c_arm=on

# Reboot
sudo reboot

# Verify
ls /dev/i2c-*
i2cdetect -y 1
```

Permission Denied on GPIO **Problem**: `RuntimeError: Cannot access GPIO`

Solution:

```
# Add user to gpio group
sudo usermod -a -G gpio $USER

# Or run service as root (default in service file)
# Service already configured to run as root
```

InfluxDB Connection Failed **Problem**: `ConnectionError: Unable to connect to InfluxDB`

Solution:

```
# Check InfluxDB is running
curl http://192.168.1.10:8086/ping

# Verify configuration
python scripts/config_tool.py --get database.influxdb_v1.host
python scripts/config_tool.py --get database.influxdb_v1.port
python scripts/config_tool.py --get database.influxdb_v1.database
```

```
# Test connection manually
python -c "import requests; print(requests.get('http://192.168.1.10:8086/ping'))"

# Check firewall on InfluxDB server
# InfluxDB server should allow port 8086
```

Database Not Found **Problem**: `database not found: filamentbox`

Solution:

```
# Create database on InfluxDB server
influx
> CREATE DATABASE filamentbox
> SHOW DATABASES
> exit

# Or via HTTP API
curl -X POST 'http://192.168.1.10:8086/query' \
  --data-urlencode "q=CREATE DATABASE filamentbox"
```

Service Issues

Service Won't Start **Problem**: `Failed to start filamentbox.service`

Diagnosis:

```
# Check detailed status
sudo systemctl status filamentbox.service -l

# View logs
sudo journalctl -u filamentbox.service -n 50

# Check for configuration errors
cd /opt/filamentcontrol
source filamentcontrol/bin/activate
python -m filamentbox.main --debug
```

Common causes: - Missing config database (filamentbox.db) - Missing encryption key (.config_key or Vault unavailable) - Python dependencies not installed - Incorrect file paths in service file - Database connection failures

Service Keeps Restarting **Problem**: Service enters restart loop

Diagnosis:

```
# Watch logs in real-time
sudo journalctl -u filamentbox.service -f

# Check for:
# - Sensor read errors
# - InfluxDB connection failures
# - Configuration errors
# - Python exceptions
```

Solutions: - Fix configuration errors - Ensure sensor is connected properly - Verify InfluxDB is accessible - Check retry/backoff settings

Web UI Not Accessible **Problem**: Cannot access web UI at http://PI_IP:5000

Diagnosis:

```
# Check service status
sudo systemctl status filamentbox.service (v2.0+ - webui integrated)

# Check if port is listening
sudo netstat -tlnp | grep 5000
# or
sudo ss -tlnp | grep 5000

# Check firewall
sudo ufw status
# Should show: 5000/tcp ALLOW
```

Solutions:

```
# Ensure service is running
sudo systemctl start filamentbox.service (v2.0+ - webui integrated)

# Check Flask dependencies
source filamentcontrol/bin/activate
pip list | grep -i flask

# Open firewall port
sudo ufw allow 5000/tcp

# Test locally first
curl http://localhost:5000
```

Sensor Issues

BME280 Not Detected **Problem**: `RuntimeError: BME280 not found on I2C bus`

Diagnosis:

```
# Check I2C is enabled
ls /dev/i2c-*

# Scan for devices
i2cdetect -y 1
# BME280 should appear at 0x76 or 0x77
```

Solutions: - Verify wiring connections - Check sensor power (3.3V) - Try different I2C address (some BME280 modules use 0x77) - Test with different I2C cable if using long wires

DHT22 Timeouts **Problem**: `RuntimeError: Timeout waiting for DHT22 response`

Solutions:

```
# Modify GPIO pin via setup.sh
sudo ./install/setup.sh
# Navigate to sensor configuration, update gpio_pin

# Check wiring
# Ensure 10kohm pull-up resistor between DATA and VCC

# Test sensor with simple script
```

```
python -c "
import adafruit_dht
import board
dht = adafruit_dht.DHT22(board.D4)
print(f'Temp: {dht.temperature}C, Humidity: {dht.humidity}%')
"
```

Inconsistent Readings **Problem**: Sensor values fluctuate wildly or show impossible values

Solutions: - Check sensor placement (away from heat sources, direct airflow) - Verify power supply stability (quality power adapter) - Add capacitor near sensor (100uF) for power filtering - Increase read_interval in configuration - Check for loose connections

Data Collection Issues

No Data in InfluxDB **Problem**: Application runs but no data appears in database

Diagnosis:

```
# Enable debug mode
sudo systemctl stop filamentbox.service
cd /opt/filamentcontrol
source filamentcontrol/bin/activate
python -m filamentbox.main --debug

# Watch for batch writes in debug output
# Should see: "DEBUG - Batch ready for write (N points)"
```

Solutions: - Verify database credentials using config tool - Check database exists (for InfluxDB: `influx -execute 'SHOW DATABASES'`) - Verify network connectivity to database server - Check for write permission errors in database logs - Verify encryption key is loaded (check .config_key or Vault)

Old Data Not Recovered **Problem**: Persisted batches not sent after restart

Diagnosis:

```
# Check for persisted data
cd /opt/filamentcontrol
ls -lh unsent_batches.db

# Check database contents
sqlite3 unsent_batches.db "SELECT COUNT(*) FROM batches;"

# Enable debug mode to see recovery process
python -m filamentbox.main --debug
# Should see: "INFO - Loaded N unsent batches from SQLite"
```

Solutions: - Verify persistence.db_path using config tool - Check file permissions on unsent_batches.db - Ensure database is accessible before starting service

Control Issues

Relay Not Switching **Problem**: Heater or fan relay doesn't activate

Diagnosis:

```
# Check control configuration via setup.sh
sudo ./install/setup.sh
# View heating_control and humidity_control settings
# Monitor control decisions
sudo journalctl -u filamentbox.service -f | grep -i "heater\|fan"

# Test relay manually
python -c "
from gpiozero import LED
relay = LED(16)  # or 20 for fan
relay.on()
import time
time.sleep(2)
relay.off()
"
```

Solutions: - Verify control is enabled: `python scripts/config_tool.py --get heating_control.enabled` - Check GPIO pin numbers (BCM numbering) - Test relay module with external power - Verify relay module is compatible (active high/low) - Check wiring and power supply to relay

Relay Cycling Too Frequently **Problem**: Heater or fan switches on/off rapidly

Solutions:

```
# Increase hysteresis gap using setup.sh
sudo ./install/setup.sh

# Navigate to heating_control or humidity_control
# Increase gap between min and max values:
#   heating: min_temp_c = 18.0, max_temp_c = 23.0 (5C gap)
#   humidity: min_humidity = 35.0, max_humidity = 60.0 (25% gap)
#   check_interval = 30 seconds (reduce frequency)

# Restart service to apply changes
sudo systemctl restart filamentbox.service
```

Debugging Tools

Enable Debug Logging

```
# Stop service
sudo systemctl stop filamentbox.service

# Run manually with debug
cd /opt/filamentcontrol
source filamentcontrol/bin/activate
python -m filamentbox.main --debug

# Debug output shows:
# - Sensor readings
# - Control decisions
# - Batch preparations
# - InfluxDB writes
# - Retry attempts
# - Error details
```

Check System Resources

```
# CPU and memory usage
htop

# Disk space
df -h

# Service resource usage
systemctl status filamentbox.service
# Shows: memory usage, CPU time, uptime

# Detailed service info
sudo systemd-cgtop
# Shows real-time resource usage per service
```

Network Diagnostics

```
# Test InfluxDB connectivity
curl -v http://192.168.1.10:8086/ping

# Test with authentication
curl -u admin:password http://192.168.1.10:8086/ping

# Check database list
curl -u admin:password http://192.168.1.10:8086/query?q=SHOW%20DATABASES

# Monitor network traffic
sudo tcpdump -i any port 8086 -A
```

Uninstallation

Complete Removal

```
# Stop and disable services
sudo systemctl stop filamentbox.service (v2.0+ - webui integrated)
sudo systemctl stop filamentbox.service
sudo systemctl disable filamentbox.service (v2.0+ - webui integrated)
sudo systemctl disable filamentbox.service

# Remove service files
sudo rm /etc/systemd/system/filamentbox.service
sudo rm /etc/systemd/system/filamentbox.service (v2.0+ - webui integrated)

# Reload systemd
sudo systemctl daemon-reload
sudo systemctl reset-failed

# Remove nginx configuration
sudo rm /etc/nginx/sites-enabled/filamentbox
sudo rm /etc/nginx/sites-available/filamentbox
# Or for Docker nginx:
sudo rm /etc/nginx/conf.d/filamentbox.conf
sudo systemctl reload nginx
```

```
# Remove installation directory
sudo rm -rf /opt/filamentcontrol

# Optional: Remove system packages (if not used by other software)
# sudo apt remove python3.13 python3.13-venv i2c-tools
```

Preserve Configuration

To keep configuration for future reinstallation:

```
# Backup configuration and data
mkdir -p ~/filamentbox-backup
cp /opt/filamentcontrol/filamentbox.db ~/filamentbox-backup/
cp /opt/filamentcontrol/.config_key ~/filamentbox-backup/
cp /opt/filamentcontrol/unsent_batches.db ~/filamentbox-backup/ 2>/dev/null

# Backup Vault configuration if used
cp /opt/filamentcontrol/vault_config.json ~/filamentbox-backup/ 2>/dev/null

# Then proceed with removal
# ...

# Restore later
sudo cp ~/filamentbox-backup/filamentbox.db /opt/filamentcontrol/
sudo cp ~/filamentbox-backup/.config_key /opt/filamentcontrol/
sudo chmod 600 /opt/filamentcontrol/.config_key
```

Partial Removal

Remove only web UI but keep main application:

```
# Stop and disable web UI service
sudo systemctl stop filamentbox.service (v2.0+ - webui integrated)
sudo systemctl disable filamentbox.service (v2.0+ - webui integrated)

# Remove service file
sudo rm /etc/systemd/system/filamentbox.service (v2.0+ - webui integrated)

# Remove nginx configuration
sudo rm /etc/nginx/sites-enabled/filamentbox
sudo rm /etc/nginx/sites-available/filamentbox
sudo systemctl reload nginx

# Reload systemd
sudo systemctl daemon-reload

# Main application continues running
sudo systemctl status filamentbox.service
```

Additional Resources

Documentation

- **Main README**: README.md - Project overview and quick start
- **Module Documentation**: filamentbox/README.md - Architecture and components

- **Testing Guide**: tests/README.md - Test documentation
- **Web UI API**: webui/README.md - REST API documentation
- **Changelog**: CHANGELOG.md - Version history

Installation Scripts

- **Master Installer**: `install.sh` - Main entry point for installation and updates
  - Handles directory setup, venv creation, dependencies
  - Calls setup.sh for configuration
  - Installs systemd services
- **Configuration Manager**: `setup.sh` - Encrypted configuration setup and management
  - Generates encryption keys
  - Configures Vault integration
  - Interactive database and sensor configuration
  - Generates service files with dynamic paths
- **Service Installers**:
  - `install_service.sh` - Main application service
  - `install_webui_service.sh` - Web UI service

Support

- **GitHub Issues**: https://github.com/jdelgado-dtlabs/filamentenvmonitor/issues
- **GitHub Discussions**: https://github.com/jdelgado-dtlabs/filamentenvmonitor/discussions

Useful Commands Reference

```
# Installation
sudo ./install/install.sh                  # Master installer
sudo ./install/setup.sh                    # Configuration setup

# Service management
sudo systemctl status filamentbox.service    # Check status
sudo systemctl restart filamentbox.service  # Restart service
sudo journalctl -u filamentbox.service -f    # View logs

# Configuration management
sudo ./install/setup.sh                    # Interactive config

# Manual testing
source /opt/filamentcontrol/filamentcontrol/bin/activate
python -m filamentbox.main --debug         # Debug mode
python filamentbox_cli.py                  # CLI interface
python webui/webui_server.py               # Web UI server

# Updates
git pull origin master                     # Pull latest code
sudo ./install/install_service.sh          # Update services

# Diagnostics
i2cdetect -y 1                            # Scan I2C bus
sudo systemctl status filamentbox.service -l # Detailed status
sudo journalctl -u filamentbox.service -n 100 # Last 100 log lines
```

Version: 1.6.0
Last Updated: December 2025
Maintained by: FilamentBox Development Team