BMS BACnet Documentation ======================

Version: 1.0 Date: 2025-12-24

Overview -------- This document provides: - Developer reference for the BACnet-specific modules: - `bms_bacnet_server_bacnet.py` (BACnet server) - `bms_bacnet_client_bacnet.py` (BACnet client) - `bms_bacnet_demo_bacnet.py` (end-to-end demo) - Usage and tutorial for running the simulator on the same machine and across networked machines - Database storage behavior used by the client - Troubleshooting and best practices

Recommended Python runtime ---------- - Recommended: Python 3.11 (bacpypes compatibility tested up to 3.11). Python 3.12 may not be supported by some bacpypes releases due to removal of `asyncore` and other changes.

Files described ---------- 1) `bms_bacnet_server_bacnet.py` — BACnet server

Purpose - Expose the simulated BMS device sensor values as BACnet objects via `bacpypes`. - Maps each sensor to an `AnalogValueObject` (temperature, humidity, pressure, CO2, occupancy). - Updates each object's `presentValue` from the running BMS simulator (`BMSDevice.get_sensor_data()`).

Key classes/usage - `BACnetBMSServer(device, address='127.0.0.1/47808', device_id=12345)` - `start(poll_interval=1.0)` — starts bacpypes core in background and an update thread that polls the simulator at `poll_interval` seconds. - `stop()` — stops the update thread and attempts to stop bacpypes core.

Important details - Requires `bacpypes` installed and a compatible Python version (3.11 recommended). - The server registers 5 analogValue objects with instance numbers 1-5 mapping to sensors: - 1: temperature - 2: humidity - 3: pressure - 4: co2_level - 5: occupancy - The analog object `presentValue` is updated every `poll_interval` seconds by reading `device.get_sensor_data()`.

2) `bms_bacnet_client_bacnet.py` — BACnet client

Purpose - Minimal `bacpypes` client wrapper to perform `ReadProperty` requests (reads `presentValue`) from remote analogValue objects. - Automatically stores each read into an SQL database (SQLite by default) via `DBHandler`.

Key classes/usage - `BACnetBMSClient(local_device_id=999, local_address='0.0.0.0/47809', db_config=None)` - `start()` — starts bacpypes core in a background thread. - `stop()` — stops bacpypes core. - `read_analog(target_address, object_type, instance, timeout=2.0)` — synchronous read of the `presentValue` property on the remote object. Returns bacpypes response (and also writes a row to the configured DB).

Database integration (DBHandler) - Default database: SQLite file `bms_bacnet.db` in the working directory. - Default table: `bms_readings` - Columns: `id`, `timestamp`, `device`, `object_type`, `instance`, `sensor_name`, `value` - `DBHandler` supports two `db_type` values: `sqlite` (default) and `postgres` (requires `psycopg2` and valid DB credentials). - When `read_analog` returns a value, the client attempts to convert it to `float` and insert it as the `value`; if conversion fails the string representation is stored.

Examples - Default (SQLite) client usage:

from bms_bacnet_client_bacnet import BACnetBMSClient

client = BACnetBMSClient() client.start() val = client.read_analog('127.0.0.1/47808', 'analogValue', 1) print('Read result:', val) client.stop()

- Explicit DB config example (Postgres):

db_cfg = { 'db_type': 'postgres', 'database': 'bacnetdb', 'user': 'bacnet_user', 'password': 'secret', 'host': '192.168.1.10', 'port': 5432, 'table': 'bms_readings' } client = BACnetBMSClient(db_config=db_cfg)

3) `bms_bacnet_demo_bacnet.py` — end-to-end demo

Purpose - Convenience script that starts a `BMSDevice` simulator, the `BACnetBMSServer`, and a `BACnetBMSClient` that reads the temperature (analogValue, instance 1) 5 times. - Intended for local single-host validation.

Usage - Run the script directly (after installing dependencies and creating a simulator file):

C:/Python311/python.exe bms_bacnet_demo_bacnet.py

Tutorials --------- A. Run everything on the same machine (recommended for first test)

1. Create a venv and install requirements for BACnet:

cd c:\Users\hamid\pyCode C:/Python311/python.exe -m venv .venv-bacnet .\.venv-bacnet\Scripts\Activate.ps1 pip install -U pip pip install -r requirements_bacnet.txt

also ensure your bms_simulator.py, bms_bacnet_server_bacnet.py, bms_bacnet_client_bacnet.py, bms_bacnet_demo_bacnet.py are in the folder

2. Run the demo (single command):

C:/Python311/python.exe bms_bacnet_demo_bacnet.py

3. Observe output like:

Read temperature (analogValue,1): 22.14 Read temperature (analogValue,1): 22.01 ...

4. Inspect database (SQLite default) in working folder: `bms_bacnet.db`. - You can view using sqlite3 CLI or a GUI tool.

B. Run server and client on separate machines (networked)

Assume host A (server) has IP `192.168.1.10`, host B (client) has IP `192.168.1.11`.

1. On host A (server): - Ensure Python 3.11 and `bacpypes` installed. - Start server with address binding to host A's interface and correct transport port. Example: `192.168.1.10/47808`.

on host A

from bms_simulator import BMSDevice from bms_bacnet_server_bacnet import BACnetBMSServer

dev = BMSDevice('BACNET-DEV1', 'Floor 1') dev.start_simulation() server = BACnetBMSServer(dev, address='192.168.1.10/47808', device_id=4001) server.start()

keep running

2. On host B (client): - Ensure Python 3.11 and `bacpypes` installed. - Use the client's `read_analog` pointing to `192.168.1.10/47808`.

from bms_bacnet_client_bacnet import BACnetBMSClient

client = BACnetBMSClient(local_device_id=999, local_address='0.0.0.0/47809') client.start() val = client.read_analog('192.168.1.10/47808', 'analogValue', 1) print('Remote temperature:', val) client.stop()

Networking notes - Ensure firewall rules on host A allow the chosen UDP/TCP transport and port (bacpypes example uses BIP/UDP-style addresses like `ip/port`). - Use stable IPs or hostnames. - For NAT or cross-network communications, configure routing and firewall accordingly.

Database notes - The client writes readings on each successful `read_analog` call. - For SQLite: concurrent writes from multiple processes may be serialized or cause locks — for multi-client writes to a shared DB prefer PostgreSQL.

Troubleshooting ---------- - "ModuleNotFoundError: bacpypes": verify virtualenv is active and `pip install bacpypes` completed. - "Connection refused" or "No response": ensure server is running and address/port match; verify firewall rules. - "Port already in use": pick a different port and update server/client addresses accordingly. - Database insertion errors: check DB credentials and that `psycopg2` is installed for Postgres.

Converting this doc to PDF ---------- A small helper script `generate_bacnet_pdf.py` is provided to convert this Markdown into a simple PDF (requires `reportlab`). See the script and instructions in the project root.

License & credits ---------- This documentation accompanies the BMS simulator project and is provided as part of the workspace.

End of document