THIẾT KẾ IoT GATEWAY CHO HỆ THỐNG GIÁM SÁT VÀ QUẢN LÝ TỦ ĐIỆN HẠ THẾ

DESIGN AN IoT GATEWAY FOR MONITORING AND MANAGING SYSTEM OF LOW-VOLTAGE DISTRIBUTION CABINET

Lê Anh Ngọc, Nguyễn Khánh Tùng

Trường Đại học Điện lực

Ngày nhận bài: 16/06/2020, Ngày chấp nhận đăng: 27/08/2020, Phản biện: TS. Nguyễn Hữu Quỳnh

Tóm tắt:

Bài báo đề xuất phương pháp thiết kế và cài đặt IoT Gateway để ứng dụng cho hệ thống giám sát và điều khiển tủ điện hạ thế qua mạng Internet. Giải pháp gateway sử dụng hệ điều hành mở và phần cứng mở Single Board Computer (SBC) hỗ trợ nhiều giao tiếp mạng khác nhau, cho phép giao tiếp được các thiết bị khác nhau trong tủ điện hạ thế và Internet. Giải pháp gateway này do đó có chi phí thấp, có khả năng dễ dàng mở rộng phần mềm dựa trên hệ điều hành mã nguồn mở Linux, đồng thời cho phép dễ dàng cài đặt nhiều kiến trúc giao thức cho các công nghệ kết nối khác nhau. Kết quả cải đặt và thử nghiệm gateway cho hai giao thức MODBUS và MQTT cho kết quả đáp ứng yêu cầu thực tế đặt ra.

Từ khóa:

Internet vạn vật (IoT), gateway, SCADA, Modbus RTU, MQTT, tủ điện hạ thế.

Abstract:

This paper proposed a method to design and implement an IoT Gateway to apply for monitoring and controlling low-voltage distribution cabinet via the Internet. This gateway uss an open source operating system and Single Board Computer (SBC) that supports various network interfaces, allowing communication of various devices in low-voltage distribution cabinet and the Internet. This gateway solution is therefore low-cost, likely easy to extendsoftware based on the open source Linux operating system, and easy to implement network protocol architectures. Results of implementing and testing gateway for MODBUS and MQTT protocols give outcomes that meet practical requirements.

Keywords:

Internet of Thing (IoT), gateway, SCADA, modbus RTU, MQTT, low-voltage distribution cabine.

1. GIỚI THIỆU CHUNG

Hàng năm, lưới điện phân phối được đầu tư phát triển không ngừng để đáp ứng nhu cầu cung cấp điện cho khách hàng. Hệ thống quản lý tủ điện hạ thế có chức năng thu thập dữ liệu trên lưới điện, giám sát các trạng thái các thiết bị và điều khiển tự động tại các điểm nút đóng/cắt quan trọng trên lưới, từ đó tăng hiệu quả vận hành, giảm tổn thất lưới điện, giảm sự cố, tăng tính an toàn trong vận hành, tạo tiền đề để phát triển lưới điện thông minh [1].

Các tủ điện hạ thế được lắp đặt ở nhiều nơi trong quận/huyện, với số lượng từ 50-100 tủ trong một khu vực có bán kính tầm 3-11 km. Nhân viên vận hành phải đi tận hiện trường để thao tác đóng/cắt điện. Trong mỗi tủ điện có các thành phần (hình 1):

- Các bộ đo điện (Power Meter-PM) để đo các thông số U, I, cosØ;
- Các cảm biến nhiệt độ, độ ẩm, trạng thái đóng mở cửa.
- Máy đóng cắt để thực hiện đóng cắt điện.

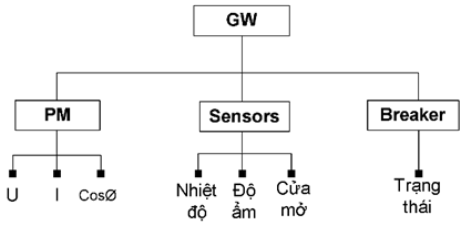
Hình 1. Tủ điện hạ thế có PM và máy cắt

Những khó khăn khi quản lý hệ thống tủ điện:

- Số lượng tủ điện/trạm hạ thế lớn, phân bố rải rác, nhưng đang vận hành thủ công: cần nhân viên trực tiếp đi đến từng trạm, ghi sổ sách.
- Các sự cố điện (mất điện, quá dòng, quá áp…), mất cắp, cháy nổ,… người vận hành không giám sát được thường xuyên.

Yêu cầu bài toán đặt ra là cần thiết kế một thiết bị gateway để thu thập các thông số cảm biến và hoạt động của tủ điện và chuyển tiếp qua Internet tới trung tâm vận hành nhằm mục đích để phân tích, cảnh báo và điều khiển máy cắt từ xa. Bên cạnh đó gateway phải có chi phí thấp và khả năng dễ dàng mở rộng kiến trúc giao thức cho các kết nối mạng khác nhau trong tương lai.

Để giải quyết vấn đề này, gateway phải có khả năng giao tiếp với PM để đọc các thông số dòng điện, giao tiếp với các cảm biến: nhiệt độ, độ ẩm, đóng/mở cửa, giao tiếp với máy cắt (breaker) để đọc trạng thái và điều khiển. Trạng thái của máy cắt (ON/OFF) được thể hiện ở tiếp điểm, cần kết nối gateway với tiếp điểm để lấy được trạng thái, từ đó biết được tủ đang có điện hay mất điện (hình 2). Gateway sau khi đọc được các thông số này sẽ gửi lên máy chủ bằng giao thức MQTT để xử lý, lưu trữ. Dựa trên trạng thái máy cắt, gateway có thể nhận lệnh điều khiển từ trung tâm để đóng/cắt điện.



Hình 2. Giao tiếp của gateway với các thành phần của tủ điện hạ thế

Chi tiết các thông số cần thu thập được phân loại trong các bảng 1, 2 và 3.

Bảng 1. Các thông số trên PM

Thông số	Tên	Mục đích thu thập	Tần suất lấy mẫu
U	Điện áp	Biết điện áp tức thời	15 s/lần
I	Cường độ dòng điện	Biết cường độ dòng tức thời	15 s/lần
CosØ	Độ lệch pha	Biết độ lệch pha	15 s/lần

Bảng 2. Các thông số trên cảm biến

Thông số	Mục đích thu thập	Tần suất lấy mẫu
Độ ẩm	Biết độ ẩm của tủ	5 phút/lần
Nhiệt độ	Biết nhiệt độ tại tủ	5 phút /lần
Trạng thái đóng/mở cửa tủ điện	An ninh (xem có bị trộm cắp không)	Mỗi lần thay đổi trạng thái

Bảng 3. Các thông số trên máy cắt

Thông số	Mục đích thu thập	Tần suất lấy mẫu
Trạng thái máy cắt (ON/OFF)	Biết đang có điện hay mất điện	Mỗi lần thay đổi trạng thái

2. MỘT SỐ CÁC GIẢI PHÁP THIẾT KẾ IoT GATEWAY

Nhiều nghiên cứu về giải pháp IoT SCADA đã được cộng đồng các nhà khoa học và các công ty công nghệ quan tâm gần đây. Trong bài báo [2], các tác giả đã đề xuất xây dựng hệ thống SCADA có chi phí thấp, sử dụng thiết bị ESP32 OLED để thu thập thông số hiện trường và gửi qua Internet sử dụng giao thức MQTT. Tuy nhiên thiết bị trên hạn chế về năng lực phần cứng và khả năng mở rộng do không có hệ điều hành.

Tác giả Shopov [3] đã đề xuất kiến trúc IoT gateway dựa trên hệ điều hành mã nguồn mở Ubuntu để thực hiện một tập hợp các công việc bao gồm giao tiếp với các thiết bị điện tại hiện trường, truyền thông với đám mây. Hạn chế của đề xuất này là không đề cập đến giả thiết về kiến trúc phần cứng, do vậy khả năng mở rộng trong thiết kế phần mềm chưa thật sự rõ ràng.

Các tác giả trong [4] đã đề xuất một giải pháp thiết kế và cài đặt IoT gateway cho giám sát và điều khiển bể bơi. Giải pháp này sử dụng máy tính nhúng Raspberry chạy hệ điều hành mở Raspberian đóng vai trò gateway, trong khi Arduino Uno như là một phần cứng mở rộng để kết nối các cảm biến. Giải pháp này do đó làm tăng độ phức tạp của hệ thống và độ trễ xử lý.

Dragan Mlakić và các tác giả [5] đã đề xuất giải pháp phần cứng và phần mềm cho IoT gateway nhằm thu thập các thông số của tủ điện hạ thế, với ưu điểm là chi phí thấp, tuy nhiên phần cứng sử dụng là mạch Arduino UNO R3 có rất nhiều hạn chế trong việc hỗ trợ lập trình và khả năng xử lý.

Mohammad Hossein Yaghmaee và các tác giả [7] đưa ra đề xuất giải pháp quản lí năng lượng hiệu quả dựa trên điện toán sương mù với home gateway sử dụng giao thức HTTP và CoAP trên nền tảng phần cứng Raspberry. Tuy nhiên, các kết quả phân tích trong [8] cho thấy được ưu điểm của giao thức MQTT so với các giao thức khác như HTTP, CoAP.

Các kết quả nghiên cứu trong [8-10] sử dụng giao thức IEC 104 cho việc thu thập dữ liệu từ các tủ điện hạ thế, tuy nhiên kiến trúc truyền thông này đòi hỏi chi phí triển khai hệ thống tương đối cao.

Để thiết kế IoT gateway cho bài toán tủ điện hạ thế, ngoài việc phải có khả năng giao tiếp được các thiết bị khác nhau trong tủ điện hạ thế và internet, gateway phải có chi phí thấp và tính dễ dàng mở rộng. Dựa trên sự phân tích ưu nhược điểm các giải pháp thiết kế gateway ở trên cho thấy giải pháp phù hợp là sử dụng

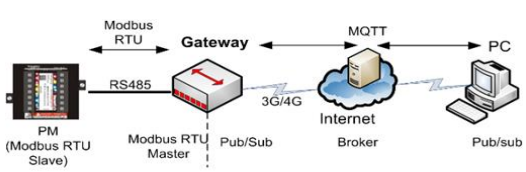
phần cứng mở Single Board Computer (SBC) hỗ trợ nhiều giao tiếp mạng khác nhau, chạy hệ điều hành mã nguồn mở để dễ dàng tùy biến. Giải pháp này cho phép dễ dàng cài đặt nhiều kiến trúc giao thức khác nhau trong đó có giao thức MODBUS và MQTT.

3. THIẾT KẾ GATEWAY

Phần này đề cập đến giải pháp thiết kế IoT gateway cho hệ thống quản lí tủ điện hạ thế. Chúng tôi đã sử dụng giải pháp phần cứng mở Raspberry, trên đó chạy hệ điều hành Raspbian mã nguồn mở nhân Linux. Gateway hỗ trợ nhiều giao tiếp mạng không dây và các kết nối có dây. Đặc biệt gateway có các chân giao tiếp GPIO analog và digital có thể lập trình được. Do gateway chạy hệ điều hành nhân Linux, nên hỗ trợ các ngôn ngữ lập trình khác nhau như C++, Python. Bên cạnh đó, gateway cho phép dễ dàng cài đặt các giao thức khác nhau trong đó có giao thức MODBUS và MQTT.

3.1. Thiết kế giao tiếp với PM thông qua giao thức MODBUS RTU

Các PM thường được nhà sản xuất hỗ trợ giao tiếp vật lý RS485 sử dụng giao thức MODBUS RTU [11], đóng vai trò là slave, lưu các giá trị *U*, *I*, cosØ tại các thanh ghi. Do đó gateway cần hỗ trợ giao tiếp RS485, lập trình cài đặt chức năng MODBUS master (hình 3).



Hình 3. Kiến trúc hệ thống với IoT gateway

Slave Address	Function Code	Data	CRC
1 byte	1 byte	0 up to 252 byte(s)	2 bytes (CRC Low, CRC Hi)

Hình 4. Cấu trúc bản tin Modbus RTU

Gateway được lập trình để gửi bản tin MODBUS request tới slave yêu cầu đọc giá trị thanh ghi và nhận về bản tin reply chứa các giá trị cần thu thập. Cấu trúc một bản tin modbus như hình 4, trong đó:

- *Slave address:* là số ID của thiết bị slave, có giá trị từ 1-255, do người sử dụng cấu hình trên PM. Trong một tủ điện có thể đấu nối tiếp các PM theo topo dạng bus, mỗi PM cần có số ID khác nhau.
- *Function code*: chứa lệnh yêu cầu mà master gửi cho slave. Thông thường mã hàm master yêu cầu đọc dữ liệu trên thanh ghi của slave là 03.
- *Data*: là dữ liệu trao đổi giữa master và slave. Trong bản tin request, phần data phải có địa chỉ thanh ghi trên slave, và số lượng thanh ghi cần đọc. Trong bản tin reply, data sẽ chứa giá trị thông số điện trả về. Lưu ý kiểu dữ liệu trả về trong tài liệu mô tả PM của hãng sản xuất.
- *CRC*: mã kiểm tra lỗi.

Sau khi master gửi bản tin request tới PM, nó sẽ khởi động đồng hồ chờ slave phản hồi (response time-out). Thời gian này được cài đặt khi lập trình, phụ thuộc vào Nếu quá thời gian time-out mà slave chưa trả lời thì master sẽ thông báo lỗi và tiến hành gửi lại request. Nếu slave phản hồi về, master sẽ kiểm tra gói tin nhận được, nếu có lỗi thì sẽ thông báo. Thời gian time-out này có thể tham khảo theo khuyến nghị trong tài liệu của thiết bị PM.

Các bước cần thực hiện trên gateway để đọc thông số trên PM:

- **Bước 1**. Cài đặt các thông số kết nối cổng serial RS485 trên PM bao gồm: data bits, baud rate, parity, stop bit và địa chỉ slave. Các thông số này lấy trong tài liệu của hãng sản xuất thiết bị.
- **Bước 2.** Cài đặt vai trò modbus master bao gồm các thông số: địa chỉ của slave, function code, thời gian time-out, kiểu dữ liệu, địa chỉ thanh ghi, số lượng thanh ghi.
- **Bước 3:** Cài đặt tần suất lấy mẫu là 15 s/lần.

3.2. Thiết kế giao tiếp với các Sensor và máy cắt qua các cổng I/O

Các thông số độ ẩm, nhiệt độ trong tủ được đo bằng các cảm biến analog, đấu nối với gateway thông qua các chân I/O analog. Trạng thái máy cắt và đóng mở cửa sẽ được lấy thông qua các cảm biến digital và đấu nối với gateway thông qua các chân I/O digital.

4. THIẾT KẾ GIAO TIẾP TRUYỀN THÔNG SỬ DỤNG GIAO THỨC MQTT

Phần này trình bày về phương pháp thiết kế và cài đặt giao thức MQTT trên gateway. Bên cạnh đó các tham số về chất lượng dịch vụ (QoS) khác nhau được giới thiệu và cài đặt cho giao thức MQTT nhằm đáp ứng các yêu cầu về thời gian thực.

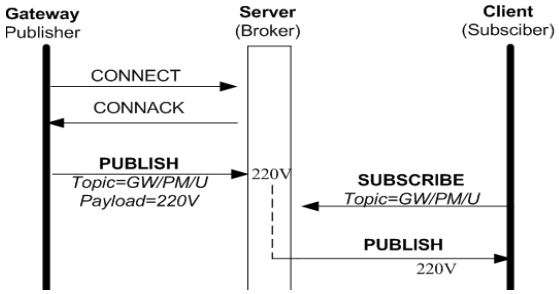
4.1. Giao thức MQTT

Phía trung tâm vận hành được trang bị máy chủ để nhận các yêu cầu kết nối từ gateway, xử lý và lưu trữ các thông số điện. Tại gateway, để gửi các thông số tới máy chủ, cần lựa chọn giao thức truyền thông có đặc điểm nhanh, hướng sự kiện và tin cậy. Trong bài báo sẽ lựa chọn giao thức MQTT (Message Queuing Telemetry Transport) - giao thức gửi dạng Publish-Subscribe. Giao thức MQTT sử dụng giao thức TCP/IP để truyền dữ liệu. MQTT được sử dụng phổ biến cho các thiết bị IoT (Internet of Things). Lý do lựa chọn MQTT là vì:

- MQTT ban đầu được thiết kế cho hệ thống thu thập dữ liệu và điều khiển từ xa (SCADA). MQTT đã trở nên phổ biến gần đây do sự phát triển của IoT.
- Là giao thức hướng sự kiện theo mô hình publish/subscribe (xuất bản - theo dõi), phù hợp cho việc truyền thông theo định kỳ.
- Thủ tục truyền tin nhanh, độ tin cậy cao thông qua cơ chế MQTT QoS.

Kiến trúc mức cao của MQTT gồm hai phần chính là broker và clients. Client có thể là publisher (gửi tin) hoặc subscriber (nhận tin). Broker có vai trò như một hub trung tâm, làm nhiệm vụ kết nối các client bằng cách nhận các bản tin từ publisher, rồi chuyển chúng đến subscriber, trong một topic nào đó. Broker có thể mở rộng thêm một vài tính năng liên quan tới quá trình truyền thông như: bảo mật bản tin, lưu trữ bản tin, ghi log [12].

Hình 5 minh họa việc sử dụng giao thức MQTT để gửi thông số điện áp (U). Gateway đóng vai trò publisher, thiết lập phiên kết nối với broker bằng cách gửi bản tin CONNECT và nhận về phản hồi CONNACK. Sau đó publisher gửi bản tin PUBLISH chứa giá trị điện áp hiện thời U=220V vào topic GW/PM/U trên máy chủ broker. Máy client của người vận hành đóng vai trò subscriber, đăng ký nhận dữ liệu từ topic GW/PM/U bằng bản tin SUBSCRIBE, sẽ nhận thông tin về điện áp hiện tại.



Hình 5. Gửi các thông số dòng điện bằng giao thức MQTT

4.2. Thiết lập phiên MQTT

Gateway khi bật lên sẽ kết nối với broker thông qua bản tin CONNECT chứa một số thông tin quan trọng như sau:

- **ClientID**: là định danh của gateway. ClientID được lựa chọn dựa trên sự kết hợp:

– Serial phần cứng;

– Thời gian sản xuất ra;

– Tên GW, địa điểm lắp đặt (thay đổi được, do người dùng nhập).

- **CleanSession**: là cờ báo hiệu phiên kết nối từ gateway đến broker có lưu thông tin về gateway hay không. Nếu CleanSession=true, broker không lưu bất kỳ thông tin gì về gateway, nếu CleanSession=false, broker sẽ lưu lại một số thông tin về gateway và do đó có thể thiết lập lại phiên một cách nhanh chóng nếu trước đó có sự cố. Trên gateway, nhóm tác giả để mục này cho người dùng tùy chọn, tuy nhiên cài đặt mặc định là CleanSession=false để broker lưu thông tin về gateway.

- **KeepAlive**: là thời gian (s) dài nhất mà gateway và broker có thể duy trì kết nối mà không cần gửi gói tin MQTT. Trên giao diện gateway mục này cũng là một tùy chọn của người dùng, mặc định là 120 s.

- **Username và Password**: là thông tin sử dụng để xác thực gateway trên broker. Tài khoản được tạo trên broker và cài đặt trên gateway.

Sau khi gateway gửi bản tin CONNECT, phía broker gửi lại bản tin CONNACK, trong bản tin này có trường thông tin returnCode. Nếu trường này có giá trị bằng 0 thì phiên MQTT được thiết lập thành công.

4.3. Gửi bản tin PUBLISH

Bản tin PUBLISH mà gateway gửi lên chứa các trường sau:

- **PacketID**: là ID của gói tin.

- **TopicName**: là tên của topic trên broker. Gateway cần biết được tên các topic trên broker để gửi thông số lên đó. Bài báo đưa ra 04 topic có đường dẫn theo cấu trúc như hình 2.

– $Client_{ID}$/PM/[thông số điện];

– $Client_{ID}$/Sensors/[thông số cảm biến];

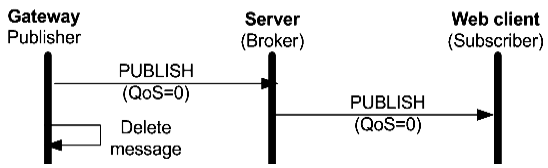
– $Client_{ID}$/Breaker/[trạng thái];

– $Client_{ID}$/Breaker/[lệnh điều khiển].

- **QoS**: xác định mức độ tin cậy khi gửi bản tin PUBLISH, mức QoS áp dụng cho gateway được phân tích sau đây.

- **RetainFlag**: cờ này được đặt giá trị bằng true để subscriber (máy người vận hành) có thể nhận thông tin từ broker ngay sau khi đăng ký.

- **Payload**: là dữ liệu đọc được từ PM, Sensors và máy cắt. Payload được lập trình mã hóa, sử dụng khóa chia sẻ giữa client và broker (pre-share key).



Hình 6. Gửi các bản tin MQTT với QoS=0

4.4. Vấn đề QoS và lưu trữ bản tin

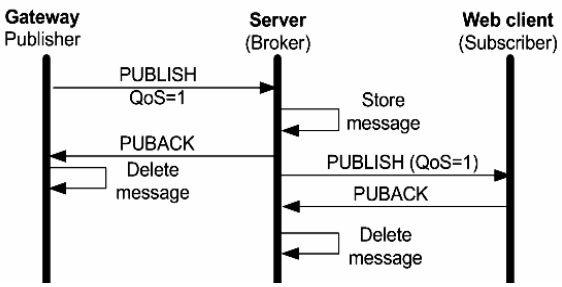
Bản tin PUBLISH có trường QoS với 3 mức là 0, 1, 2.

- **Mức 0** (hình 6): trong mức này publisher chỉ đơn giản là gửi bản tin cho broker, và sau đó xóa bản tin. Broker không gửi báo nhận về cho publisher, mà chuyển tiếp bản tin cho subscriber. Ưu điểm là thủ tục truyền tin nhanh, đơn giản. Nhược điểm là không đảm bảo tính tin cậy vì bản tin gửi đi có thể bị mất. Mặc dù MQTT dựa trên giao thức TCP, tuy nhiên trong những tình huống gateway mất kết nối 3G/4G, dẫn đến mất phiên TCP (do port nguồn trên gateway đổi khi thiết lập lại phiên TCP mới), và khi đó các gói TCP mất sẽ không được truyền lại. Quá trình thử nghiệm cho thấy mức QoS này phù hợp để:

– Truyền các thông số điện, thông số cảm biến từ gateway lên broker và từ broker về subscriber, vì đảm bảo thời gian thực.

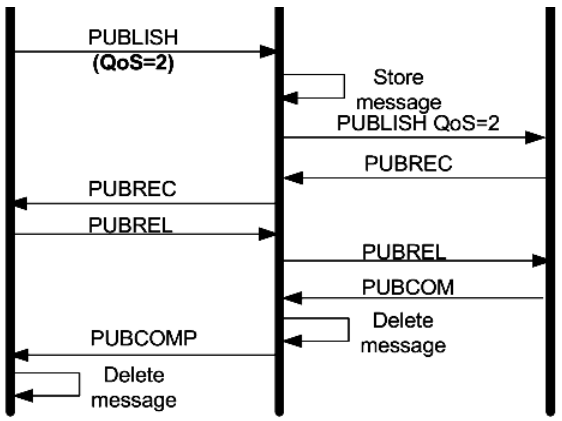
– Truyền lệnh điều khiển từ máy người vận hành lên broker, và từ broker xuống gateway.

Mức 1 (hình 7) là mức bảo đảm rằng bản tin sẽ đến đích ít nhất 1 lần bằng cơ chế báo nhận. Phía broker khi nhận được bản tin từ publisher, nó sẽ lưu lại, sau đó gửi trả về bản tin báo nhận PUBACK. Sau khi nhận được PUBACK thì publisher mới xóa bản tin đã gửi. Quá trình trao đổi bản tin giữa broker và subscriber diễn ra tương tự. Có thể xảy ra tình huống PUBACK bị mất, khi đó publisher sẽ gửi lại bản tin PUBLISH với cờ DUP (duplicate) được bật, để thông báo cho broker biết đây là bản tin gửi lại. Căn cứ vào đó phía broker sẽ xử lý gói tin có cờ DUP theo logic nghiệp vụ của ứng dụng. Ưu điểm của mức QoS này là phía broker sẽ nhận được bản tin ít nhất một lần, nhược điểm là cần xử lý những bản tin trùng lặp. Điều này có thể làm tăng công việc phía máy chủ broker khi số lượng gateway lớn (50-100 bộ) và tần suất lấy mẫu dày. Do vậy Gateway không sử dụng mức QoS=1 để gửi bản tin PUBLISH.



Hình 7. Gửi các bản tin MQTT với QoS=1

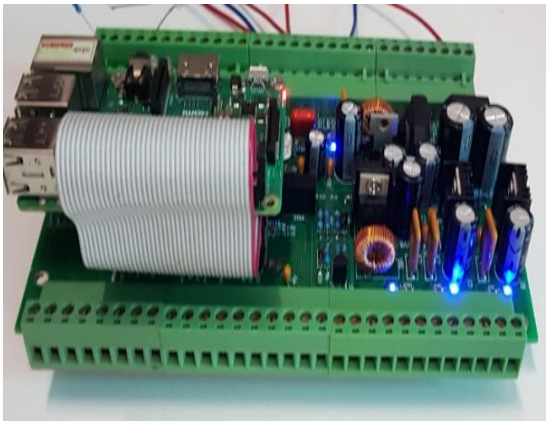
- **Mức 2** (hình 8): là mức bảo đảm rằng mỗi bản tin sẽ đến đích và đến đúng 1 lần, bằng cách dùng thêm bản tin báo nhận PUBREC, PUBREL và PUBCOMP. Chính nhờ đặc tính tin cậy này nên phù hợp để truyền trạng thái máy cắt. Nhược điểm của mức QoS này là quá trình thủ tục truyền tin phức tạp, gateway cần phải chờ nhận đủ 2 bản tin PUBREC và PUBCOM cho 1 bản tin PUBLISH gửi đi. Bản tin PUBLISH trạng thái máy cắt gửi lên cần đặt cờ Retain=true để phía máy vận hành luôn nhận được trạng thái máy cắt ngay sau khi subscribe.



Hình 8. Gửi các bản tin MQTT với QoS=2

5. CÀI ĐẶT VÀ THỬ NGHIỆM

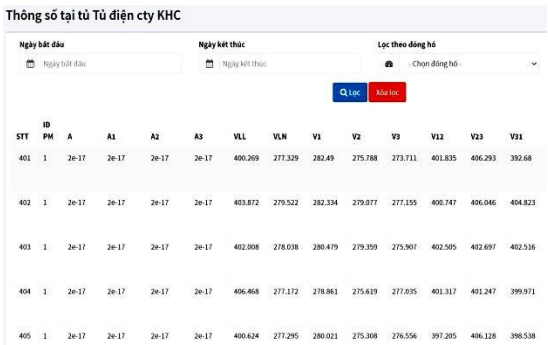
Trong bài báo này chúng tôi đã thử nghiệm với gateway là giải pháp phần cứng mở Raspberry Pi 3 Model B chạy trên hệ điều hành mã nguồn mở Raspbian nhân Linux, sử dụng bộ vi xử lý ARMv8 tốc độ 1.2 Ghz, bộ nhớ RAM 1 GB. Gateway có kích thước nhỏ, chi phí thấp, có các chân giao tiếp IO analog và digital có thể lập trình được (hình 9). Gateway đã cài đặt các giao thức khác nhau trong đó có giao thức MODBUS và MQTT. Kết quả thử nghiệm được tiến hành tại tủ điện hạ thế của Công ty TNHH KHC (địa chỉ: Phường Thanh Xuân Bắc - Quận Thanh Xuân - Hà Nội).



Hình 9. Bo mạch gateway

Kịch bản 1: *Thu thập dữ liệu từ tủ điện hạ thế của Công ty TNHH KHC*

Trong kịch bản thử nghiệm này, tại tủ điện hạ thế của Công ty TNHH KHC, gateway đọc các thông số dòng điện 3 pha từ PM và các thông số khác sử dụng giao thức Modbus đã cài đặt tại gateway. Sau đó gateway đóng gói các thông số thu thập được và gửi lên máy chủ broker sử dụng giao thức MQTT với mức ưu tiên cao nhất QoS=0.



Hình 10. Kết quả thu thập dữ liệu từ tủ điện hạ thế của Công ty TNHH KH

- **Bước 1.** Trên giao diện web tại thiết bị smartphone hoặc máy tính, Người quản trị chọn Danh sách các tủ điện hạ thế của Điện lực Quận Thanh xuân (hình 11). Kết quả cho thấy tủ điện của Công ty TNHH KHC đang có điện (thể hiện ở cột Trạng thái thực tế).

Hình 11. Trạng thái máy cắt

- **Bước 2.** Người quản trị bấm nút "Thông số" của tủ điện Công ty TNHH KHC trên giao diện web. Lúc này, trình duyệt web đóng vai trò subscriber nhận được dữ liệu từ broker gửi về và hiển thị chính xác dữ liệu thu thập được từ tủ điện hạ thế của Công ty KHC trên giao diện của người quản trị (hình 10).

Trong kịch bản thử nghiệm này, tại tủ điện hạ thế của công ty TNHH KHC.

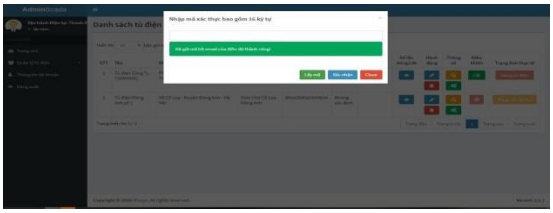
- **Bước 2.** Người quản trị bấm nút "Thông số" của tủ điện Công ty TNHH KHC trên giao diện web. Lúc này, trình

duyệt web đóng vai trò subscriber nhận được dữ liệu từ broker gửi về và hiển thị chính xác dữ liệu thu thập được từ tủ điện hạ thế của Công ty KHC trên giao diện của người quản trị (hình 10).

Kịch bản 2: *Điều khiển đóng/cắt điện*

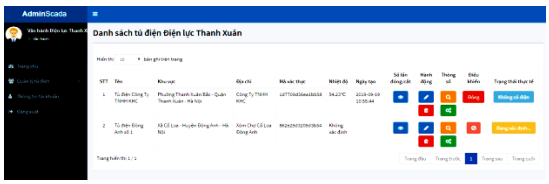
- **Bước 1.** Trên giao diện Web tại thiết bị smartphone hoặc máy tính, nhân viên vận hành chọn các tủ điện hạ thế tại địa điểm thực nghiệm là Điện lực Quận Thanh Xuân (hình 11). Kết quả cho thấy tủ điện của Công ty TNHH KHC đang có điện.

- **Bước 2.** Nhân viên vận hành bấm nút điều khiển "Cắt" điện trên một tủ điện Công ty TNHH KHC. Để đảm bảo an toàn, hệ thống thông báo cần có xác nhận cho phép đóng điện từ bên điều độ. Tại thời điểm này, một mã xác nhận từ hệ thống đã được gửi cho nhân viên điều độ để thông báo bên vận hành đang thực hiện đóng điện, cần điều độ xác nhận bằng cách gửi lại mã này cho bên vận hành (hình 12).



Hình 12. Nhập mã xác thực

- **Bước 3.** Sau khi nhập mã cho thực hiện lệnh từ bên điều độ thì lệnh đóng được thực hiện với bản tin có QoS=0. Hệ thống đã cắt điện thành công, thể hiện ở Trạng thái thực tế đã thay đổi thành Đang không có điện (hình 13).



Hình 13. Kết quả thao tác cắt điện tại tủ điện của Công ty KHC

6. KẾT LUẬN

Bài báo đề xuất phương pháp thiết kế và cài đặt IoT gateway cho hệ thống giám sát và điều khiển tủ điện hạ thế qua mạng internet. Giải pháp gateway được đề xuất sử dụng phần cứng và hệ điều hành mở cho phép hỗ trợ nhiều giao tiếp mạng khác nhau để giao tiếp được các thiết bị khác nhau trong tủ điện hạ thế và internet. Giải pháp gateway này do đó có chi phí thấp, có khả năng dễ dàng mở rộng cho nhiều kiến trúc giao thức khác nhau. Kết quả cải đặt và thử nghiệm gateway cho hai giao thức MODBUS và MQTT cho kết quả đáp ứng yêu cầu thực tế đặt ra. Trong những bài viết tiếp theo, nhóm tác giả sẽ đi sâu phân tích vấn đề tối ưu hóa xử lý thời gian thực tại gateway và vấn đề cân bằng tải của gateway trong hệ thống giám sát và điều khiển tủ điện hạ thế.

TÀI LIỆU THAM KHẢO

[1] Roger N. Anderson (2018), Smart Grid The Future of the Electric Energy System.

[2] Lawrence O. Aghenta* and M. Tariq Iqbal* (2019), Design and implementation of a low-cost, open source IoT-based SCADA system using ESP32 with OLED, ThingsBoard and MQTT protocol, AIMS Electronics and Electrical Engineering, 4(1): 57–86.

[3] M.P. Shopov (2017), IoT Gateway for Smart Metering in Electrical Power Systems - Software Architecture.

[4] André Glória et al (2017), Design and implementation of an IoT gateway to create smart environments, the 8th International Conference on Ambient Systems, Networks and Technologies.

[5] Dragan Mlakić1 et al (2019), An Open-Source Hardware/Software IED based on IoT and IEC 61850 Standard, 2nd International Colloquium on Smart Grid Metrology (SMAGRIMET).

[6] Mohammad Hossein Yaghmaee, A Fog-Based Internet of Energy Architecture for Transactive Energy Management Systems, DOI 10.1109/JIOT.2018.2805899, IEEE Internet of Things Journal.

[7] Desh Deepak Sharma (2018), The Challenges in Development of Internet of Things Based Smart Power Distribution System, 2018 5th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON).

[8] Thomas Teodorowicz (2017), Comparison of SCADA protocols and implementation of IEC 104 and MQTT in MOSAIK.

[9] Prakash Pawar* and Panduranga Vittal K (2018), Performance analysis of a smart meter node for congestion avoidance and LoS coverage, AIMS Energy, 7(3): 313–336.

[10] Panagiotis Radoglou-Grammatikis (2019), Attacking IEC-60870-5-104 SCADA Systems, 2019 IEEE World Congress on Services (SERVICES).

[11] MODBUS Protocol, www.modbus.org

[12] HiveMQ - Enterprise ready MQTT to move your IoT data, www.hivemq.com

Giới thiệu tác giả:

Tác giả Lê Anh Ngọc tốt nghiệp đại học ngành toán và tin học tại Trường Đại học Vinh và Trường Đại học Khoa học tự nhiên, Đại học Quốc gia Hà Nội các năm 1996 và 1998. Năm 2001 nhận bằng Thạc sĩ ngành công nghệ thông tin tại Trường Đại học Bách khoa Hà Nội và năm 2009 nhận bằng Tiến sĩ tại Đại học Quốc gia Kyungpook – Hàn Quốc, chuyên ngành kỹ thuật thông tin và truyền thông. Hiện nay tác giả đang công tác tại Trường Đại học Điện lực.

Hướng nghiên cứu chính: hệ thống thời gian thực, mạng truyền thông, Internet of Things, tính toán thông minh.

Tác giả Nguyễn Khánh Tùng tốt nghiệp đại học ngành công nghệ thông tin năm 2003 tại Học viện Công nghệ bưu chính viễn thông; năm 2016 nhận bằng Thạc sĩ Công nghệ thông tin, chuyên ngành hệ thống thông tin tại Trường Đại học Công nghệ - Đại học Quốc gia Hà Nội. Hiện nay tác giả đang công tác tại Trường Đại học Điện lực.

Hướng nghiên cứu chính: an ninh mạng, mạng truyền thông, Internet of Things.