PLATAFORMA IOT PARA MONITORAMENTO DE LABORATÓRIOS E EXPERIMENTOS

Alan Franco Rosa

Dissertação apresentada à Coordenação de Formação Científica do Centro Brasileiro de Pesquisas Físicas, como parte dos requisitos para obtenção do título de Mestre no programa de Mestrado Profissional em Física com ênfase em Instrumentação Científica.

Orientador: Prof. D.Sc. Nilton Alves Júnior - CBPF

Rio de Janeiro
Setembro de 2022

CBPF
Centro Brasileiro de Pesquisas Físicas
SUBORDINADA AO MCTI

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÕES

"PLATAFORMA IoT PARA MONITORAMENTO DE LABORATÓRIOS E EXPERIMENTOS"

ALAN FRANCO ROSA

Dissertação de Mestrado Profissional em Física com ênfase em Instrumentação Científica, apresentada no Centro Brasileiro de Pesquisas Físicas do Ministério da Ciência Tecnologia e Inovação. Fazendo parte da banca examinadora os seguintes professores:

Nilton Alves Júnior – Orientador/CBPF

gov.br Documento assinado digitalmente
FLAVIA COIMBRA DELICATO
Data: 05/10/2022 08:48:31-0300
Verifique em https://verificador.iti.br

Flávia Coimbra Delicato - UFF

Márcio Portes de Albuquerque - CBPF

Rio de Janeiro, 28 de setembro de 2022.

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÕES
CENTRO BRASILEIRO DE PESQUISAS FÍSICAS

PLATAFORMA IOT PARA MONITORAMENTO DE LABORATÓRIOS E EXPERIMENTOS

Alan Franco Rosa

DISSERTAÇÃO SUBMETIDA À BANCA EXAMINADORA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE NO PROGRAMA DE MESTRADO PROFISSIONAL EM FÍSICA COM ÊNFASE EM INSTRUMENTAÇÃO CIENTÍFICA DO CENTRO BRASILEIRO DE PESQUISAS FÍSICAS.

Orientador: Prof. D.Sc. Nilton Alves Júnior - CBPF

Examinada por: Prof. D.Sc. Márcio Portes de Albuquerque - CBPF
Prof. D.Sc. Flávia Coimbra Delicato - IC/UFF
Prof. D.Sc. Herman Pessoa Lima Júnior - CBPF
Prof. D.Sc. Guilherme Penello Temporão - CETUC/PUC-RJ
Prof. D.Sc. Evandro Luiz Cardoso Macedo - COPPE/UFRJ

RIO DE JANEIRO, RJ – BRASIL
SETEMBRO DE 2022

Franco Rosa, Alan

Plataforma IoT para Monitoramento de Laboratórios e Experimentos/Alan Franco Rosa. – Rio de Janeiro: CBPF, 2022.

XIII, 83 p.: il.; 29, 7cm.

Orientador: D.Sc. Nilton Alves Júnior - CBPF

Dissertação (mestrado) – CBPF/Programa de Mestrado Profissional em Física com ênfase em Instrumentação Científica, 2022.

Referências Bibliográficas: p. 65 – 72.

1. IoT. 2. Qualidade de Energia. 3. Monitoramento Elétrico. 4. Monitoramento Ambiental. I. Alves Júnior - CBPF, D.Sc. Nilton. II. Centro Brasileiro de Pesquisas Físicas, Programa de Mestrado Profissional em Física com ênfase em Instrumentação Científica. III. Título.

Agradecimentos

Gostaria de agradecer, primeiramente, aos meus pais, pela presença em todos os momentos, pelo suporte incondicional para tornar tudo isso realidade. Isto só foi possível graças a vocês! A minha companheira Suellen, por todo apoio e dedicação durante essa etapa. Ao meu irmão, Anderson, por estar sempre ali à disposição para me apoiar e a sua esposa Jaqueline.

Ao professor Nilton, pela convivência no Núcleo de Informação e Biblioteca (NIB)/CBPF, por ser para mim uma referência de sabedoria técnica e que me proporcionou a oportunidade de trabalhar no CBPF e, assim, dar vida a este projeto.

Ao David, pela parceria e pelas longas conversas no CBPF.

Ao Maurício da eletrônica/COTEC, pela convivência e aprendizado no CBPF.

Ao Marcelo Portes de Albuquerque, pelo apoio quando eu ainda era bolsista na COTEC/CBPF.

Ao pessoal do NIB do CBPF.

RESUMO

A utilização da Internet das Coisas (IoT) cresceu nos últimos anos graças ao avanço das tecnologias de comunicação sem fio, possibilitando conexões seguras e com alta velocidade. Com isso, a possibilidade de automação de processos e conexão de dispositivos passou a ser possível em praticamente qualquer lugar.

A utilização de soluções IoT como ferramenta auxiliar aos processos de aquisição de dados e monitoramento ambiental, de laboratórios e experimentos ainda é muito pouco difundida nos ambientes de pesquisa. Ainda mais se forem consideradas as formas de implementação segura dessas propostas.

Sendo assim, nesta dissertação foi desenvolvida uma plataforma IoT para monitoramento de ambientes, laboratórios e experimentos. O projeto foi desenvolvido seguindo o paradigma de *Cloud of Things* (CoT) envolvendo as camadas de nuvem, processamento de borda e de dispositivos. Na camada de nuvem foi implementado um ambiente de virtualização com alta disponibilidade, capacidade de processamento e armazenamento. Na camada de processamento de borda foi utilizado o Raspberry Pi como *gateway* IoT, com a finalidade de integrar as conexões dos sensores, através da rede sem fio, sendo a comunicação com a nuvem feita através de rede cabeada. Ainda na camada de borda também foi implementado um ambiente para visualização dos dados coletados utilizando o Grafana. Finalmente, na camada de dispositivos, foi utilizado um módulo baseado no ESP32, no qual os sensores são conectados, os dados coletados e enviados para o Raspberry Pi.

Com a preocupação de se evitar ataques cibernéticos e físicos, o *firmware* embarcado no ESP32 é criptografado.

Também foram feitos testes de desempenho, utilizando a ferramenta JMeter, para determinar o comportamento do sistema em relação a quantidade de requisições simultâneas. Nesses testes foi possível verificar que a API é capaz de responder a aproximadamente 90 requisições por segundo antes de começar a rejeitar as conexões.

A facilidade de adaptação e integração da plataforma com outros sistemas foi comprovada a partir da utilização do equipamento em diversos ambientes do CBPF.

Por fim, o projeto foi implantado seguindo o que estabelece a norma IEEE 2413-2019 dentro do laboratório contêiner do projeto Neutrinos-Angra, localizado no complexo da Usina Nuclear de Angra II.

Palavras-chave: Internet das Coisas, Monitoramento, Plataforma IoT, ESP32.

ABSTRACT

Internet of Things (IoT) has grown thanks to advances in wireless communication technologies, allowing for secure and high-speed connections. As a result, automating processes and connecting devices have become possible almost anywhere.

The use of IoT solutions as an auxiliary tool in the processes of data acquisition and environmental monitoring in laboratories and experiments is still not widespread in research environments. Even more, if considered the ways of safe implementation of these proposals.

This dissertation develops an IoT platform to operate in the monitoring of environments, laboratories, and experiments. The project was developed following the Cloud of Things (CoT) paradigm involving the cloud, edge processing, and device layers. In the cloud layer, it has been implemented a virtualization environment with high availability, processing, and storage capacity. The edge processing layer used the Raspberry Pi as an IoT gateway to integrate the connections through the network and cloud. Still in the edge layer, an environment using Grafana is used to visualize the data collected from sensors. Finally, in the device layer, a module based on ESP32 was used, in which it connects the sensors, collects the data and send them to the Raspberry Pi.

In the interest of preventing cyber and physical attacks, embedded encryption in the communication layer is used alongside firmware cryptography in order to mitigate physical attacks that aim to insert an altered firmware into ESP32.

Performance tests were also carried out, using the JMeter tool to determine the system's behavior concerning the number of simultaneous requests. In these tests, it was possible to verify that the API is capable of responding to 90 requests per second before beginning to reject connections.

As a result, it proved the ease of adapting and integrating the platform with other systems using the equipment in different CBPF environments.

Finally, the project was implemented following the IEEE 2413-2019 standard within the container laboratory of the Neutrinos-Angra project, located in the Angra II Nuclear Power Plant complex.

Keywords: Internet of Things, Monitoring, IoT Platform, ESP32.

Conteúdo

Lista de Figuras **ix**

Lista de Tabelas **xi**

Lista de Abreviaturas **xii**

1 Introdução **1**

1.1 Contextualização 1
1.2 Motivação 3
1.3 Objetivo 4
1.4 Organização da Dissertação 4

2 Revisão Bibliográfica **5**

2.1 Padronização da Arquitetura IoT 5
2.2 Aspectos Sobre o Paradigma da Internet das Coisas 6
2.3 Medidores Inteligentes de Eletricidade 7
2.4 Utilização de IoT na Condução de Experimentos 7
2.5 Utilização de IoT no Gerenciamento de Laboratórios 8
2.6 A Evolução Proporcionada pela Internet das Coisas 8
2.7 Aspectos de Segurança Para um Dispositivo IoT Utilizando o ESP32 8
2.8 Plataforma IoT 9
2.9 Requisitos da Plataforma 9

3 Materiais e Métodos **11**

3.1 Arquitetura do Projeto 11
3.2 Análise dos requisitos 12
3.3 Escolha do Material 13
3.4 Montagem do Equipamento 15
3.5 Implementação da Arquitetura de Comunicação 23
3.6 Desenvolvimento do Middleware 25
3.7 Desenvolvimento do Supervisor 27
3.8 Integração e Testes 28

4 Segurança IoT **30**
4.1 Aspectos de Segurança 30
4.2 Criptografia na comunicação 32
4.3 Criptografia do firmware 35
4.4 Criptografia Flash e Inicialização Segura 36

5 Resultados e Discussões **40**
5.1 Testes de desempenho 40
5.2 Supervisor 42
5.3 Monitoramento elétrico na Criogenia 43
5.4 Monitoramento elétrico e ambiental no Lab3I 45
5.5 Detecção da onda de choque do vulcão de Tonga 49
5.6 Aplicação no Monitoramento do ambiente de obras raras 49
5.7 Aplicação no Projeto Neutrinos Angra 50
5.8 Desafios na implementação 57

6 Conclusão **60**
6.1 Análise dos Requisitos 60
6.2 Montagem do Equipamento 61
6.3 Arquitetura de Comunicação 61
6.4 Desenvolvimento da API 61
6.5 Supervisório 61
6.6 Segurança 62
6.7 Perspectivas Futuras 62

Bibliografia **65**

A Diagramas **73**

B Descrição da API **75**

C Produção Científica **80**
C.1 Notas Técnicas do CBPF 80
C.2 Notícias 80

Lista de Figuras

2.1 Definição de "coisa" segundo a IEEE 2413-2019 [1]. 6

3.1 Microcontrolador e medidor de energia. 13
3.2 Transformador de corrente e sensor de pressão, umidade e temperatura. 14
3.3 Diagrama de conexão do sistema. 16
3.4 Exemplo do modo de operação autônoma do equipamento em teste na sala de servidores da COTEC do CBPF. 17
3.5 Fluxograma de operação do *firmware* no modo de monitoramento. . . 19
3.6 Diagrama de conexão do dispositivo desenvolvido. 21
3.7 Suporte feito utilizando a impressora 3D para acomodação dos medidores de energia. 21
3.8 Painel do equipamento com algumas das informações coletadas. . . . 22
3.9 Topologia da comunicação. 24
3.10 Detalhe da arquitetura. 25
3.11 Implementação da API. 27
3.12 Comparação dos valores medidos pelo equipamento, exibidos por fase. 29

4.1 Ataque *Man-in-the-middle*. 31
4.2 Interceptação da comunicação não criptografada (HTTP) entre o ESP32 e o servidor, utilizando o Wireshark. 33
4.3 Dados não criptografados enviados pelo ESP32 para a API. 33
4.4 Interceptação da comunicação criptografada (HTTPS) entre o ESP32 e o servidor, utilizando o Wireshark. 34
4.5 Dados criptografados através do protocolo TLSv1.2. 34
4.6 Taxonomia de uma vulnerabilidade de hardware. 35
4.7 Trecho do *firmware* não criptografado com os dados da *API Key*. . . 37
4.8 Trecho do *firmware* não criptografado com os dados de acesso Wi-Fi. 37
4.9 Fluxograma do procedimento de encriptação do *firmware*. 38
4.10 Trecho do *firmware* criptografado com os dados da *API Key*. 39
4.11 Trecho do *firmware* criptografado com os dados de acesso Wi-Fi. . . . 39

5.1 Quantidade média de requisições por segundo para até 100 dispositivos, com granularidade em minutos. 41
5.2 Frequência relativa dos tempos médios de resposta. 41
5.3 Página inicial do sistema de supervisão. 42
5.4 Gráfico do sistema para o monitoramento de fases através da representação geométrica. 43
5.5 Modo de operação autônoma do equipamento em teste na Criogenia. 44
5.6 Variação da corrente elétrica durante a operação do Chiller. 45
5.7 Equipamento instalado em um quadro de alimentação. 46
5.8 Variação da temperatura ao longo do tempo no Lab3I, CBPF. 46
5.9 Variação da umidade ao longo do tempo no Lab3I, CBPF. 47
5.10 Variação da pressão atmosférica ao longo do tempo no Lab3I, CBPF. 48
5.11 Dashboard criado no Grafana para monitoramento do painel elétrico. 48
5.12 Variação na pressão atmosférica devido a erupção do vulcão em Tonga, medido no Lab3I, CBPF. 49
5.13 Painel de monitoramento do acervo utilizando. 50
5.14 Localização do contêiner Neutrinos. (Crédito: Herman P. Lima) . . . 51
5.15 Topologia para monitoramento do contêiner. 52
5.16 Representação do local de instalação. 53
5.17 Módulo GPS NEO-M8N para sincronização da hora. 54
5.18 Desvio na hora coletada pelo GPS durante um dia. 54
5.19 Visualização das variáveis monitoradas em teste no laboratório. . . . 55
5.20 Instalação dos sensores dentro do laboratório contêiner. 56
5.21 Plataforma em funcionamento. 56
5.22 Visualização das variáveis monitoradas no ambiente de Obras Raras do CBPF. 57
5.23 Visualização das variáveis monitoradas no laboratório contêiner. . . . 57
5.24 Resposta do sensor RCWL0516 com o Wi-Fi ligado durante a medição. 58
5.25 Resposta do sensor RCWL0516 com o Wi-Fi desligado durante a medição. 59

A.1 Diagrama de Classe para o sensor PZEM 73
A.2 Diagrama de Classe para o parser agregador de sensores 74

Lista de Tabelas

3.1 Dados do sensor PZEM-004T [2]. 14
3.2 Dados do PZCT-02 [3]. 15
3.3 Dados do módulo com o BME280 [4]. 15
3.4 Tabela com as informações enviadas através da rota “medidores”. . . . 20
3.5 Tabela com as informações enviadas através da rota “medidas”. 20
3.6 Estimativa de custo para o sistema de monitoramento para uma rede trifásica. 23

Lista de Abreviaturas

AIEA	Agência Internacional de Energia Atômica, p. 51
API	*Application Programming Interface*, p. 25
BNDES	Banco Nacional de Desenvolvimento Econômico e Social, p. 2
CBPF	Centro Brasileiro de Pesquisas Físicas, p. 1, 4, 49
CCN	Catálogo Coletivo Nacional, p. 50
CEPEL	Centro de Pesquisas de Energia Elétrica, p. 1
CoT	*Cloud of Things*, p. v, 11
GPS	*Global Positioning System*, p. 53
IBICT	Instituto Brasileiro de Informação em Ciência e Tecnologia, p. 50
IDS	*Intrusion Detection System*, p. 24
IEDI	Instituto de Estudos para o Desenvolvimento Industrial, p. 2
IOC	Intelligent Operation Center, p. 27
IoLT	*Internet of Laboratory Things*, p. 8
IoT	*Internet of Things*, p. v, 1, 4, 7, 30
JSON	*JavaScript Object Notation*, p. 26
Lab3I	Laboratório de Informação, Instrumentação e IoT, p. 45
M2M	*Machine to Machine*, p. 2
MVT	*Model View Template*, p. 26
NSA	*National Security Agency*, p. 24
NTP	*Network Time Protocol*, p. 53

RTC *Real Time Clock*, p. 53

SELinux *Security Enhanced Red Hat Linux*, p. 24

SOC *System On Chip*, p. 13

WAF *Web Application Firewall*, p. 24

Capítulo 1

Introdução

Neste capítulo é contextualizado o tema Internet das Coisas (*IoT*), a motivação e o objetivo deste trabalho. Uma grande parte da inspiração para o desenvolvimento dessa dissertação veio da vivência em ambientes de pesquisa, como no Departamento de Linhas e Estações do Centro de Pesquisas de Energia Elétrica (CEPEL) e no Centro Brasileiro de Pesquisas Físicas (CBPF), onde o processo de aquisição de dados experimentais são intrínsecos às pesquisas desenvolvidas.

1.1 Contextualização

IoT pode ser, de forma simplificada, considerada como uma infraestrutura composta de equipamentos que trocam informações entre si sem a interferência humana, através de uma rede (*Network of Things*). Cada "coisa" é identificada de forma única (ID) através de seu sistema embarcado, sendo capaz de se comunicar utilizando a Internet. O sistema embarcado, também conhecido como *firmware*, é um programa capaz de executar tarefas específicas através do *hardware* para o qual ele foi desenvolvido.

De acordo com RAYES [5], a definição mais completa do termo IoT seria:

IoT é uma tecnologia que envolve **Padrões** e **Processos** que possibilitam que **Coisas** possam se conectar através da **Internet** para trocar **Dados** utilizando **Padrões** que garantem a **Interoperabilidade** e permitem, de forma completa, a **Automação** dos processos.

Nessa definição foi dado destaque a algumas palavras-chave que esclarecem que a IoT é um conjunto de tecnologias que, quando utilizadas em harmonia, executam uma tarefa. Então, é possível abstrair que a IoT envolve:

- Padrões (normas, protocolos);
- Processos;
- Coisas (sensores, atuadores);
- Internet (rede);
- Dados (informação);
- Automação (execução de tarefa, tomada de decisão).

Sendo assim, a IoT cria um novo paradigma onde objetos físicos são interconectados através de uma rede e possuem a capacidade de realizar processamento, capturar variáveis ambientais e reagir a estímulos externos [6]. Um dos maiores desafios da IoT é a heterogeneidade dos ambientes nos quais ela está inserida, tornando necessário o desenvolvimento de plataformas *middleware* capazes de prover a interoperabilidade e gerenciamento através das aplicações.

Dentro desse contexto, ainda é necessário levar em consideração que apenas recentemente houve a publicação de uma norma com o intuito de padronizar o desenvolvimento da tecnologia IoT [1], algo que até então não existia e trazia dificuldades para a definição de um modelo de desenvolvimento ideal dessas plataformas [7].

O Relatório do Plano de Ação feito a partir do estudo *Internet das Coisas: um plano de ação para o Brasil* [8], desenvolvido pelo Banco Nacional de Desenvolvimento Econômico e Social (BNDES), diz que a "Internet das Coisas é uma oportunidade única", e traz orientações sobre quais aplicações devem ser priorizadas no Brasil.

Um estudo feito pelo Instituto de Estudos para o Desenvolvimento Industrial (IEDI) [9] , publicado em 2017, ressalta que o Brasil possui grandes desafios para se adequar a essa nova evolução. Com a Indústria 4.0 muitos serviços poderão perder o sentido de existir e isso poderá criar um aumento nas taxas de desemprego, ao mesmo tempo em que exigirá um maior nível de qualificação e capacidade intelectual dos trabalhadores. Logo, o contexto sócio-econômico será um grande entrave para o desenvolvimento dessa nova evolução no Brasil.

Outro aspecto importante na utilização da IoT como objeto dessa dissertação é estimular a disseminação no ambiente de pesquisa das aplicações e recomendações que estão em voga na economia e na indústria. Com o *Plano Nacional de Internet das Coisas* [10] o governo federal pretende fomentar o desenvolvimento dessa tecnologia no Brasil e cabe ao Ministério de Ciência, Tecnologia e Inovações decidir quais são as áreas prioritárias que irão receber mais incentivos.

A aprovação do PL 6549/2019 no Senado [11], os estudos feitos e as crescentes parcerias que são firmadas em nível de governo [12] reforçam essa ideia. Somado a isso, o governo federal sancionou a Lei 14.108 [13], originária do PL 6549/2019 [11], que dá incentivos à IoT, promovendo a redução de impostos das estações que integrem sistemas de comunicação máquina a máquina (*Machine to Machine - M2M*).

Para concluir, de acordo com o professor Klaus Schwab [14], fundador do Fórum Econômico Mundial, a quarta revolução industrial é marcada pela utilização de tecnologias disruptivas que conectam todo o sistema de manufatura, tornando os processos mais econômicos, rápidos e eficientes. Segundo ele, a Indústria 4.0 pode ser definida pela utilização de:

- Robôs Autônomos;
- Simulações;
- Integração de sistemas;
- Internet das Coisas;
- Cibersegurança;
- Computação em Nuvem;
- Impressão 3D;
- Realidade Aumentada;
- Big Data.

1.2 Motivação

Durante a operação de equipamentos ligados a rede elétrica, por exemplo, os problemas associados ao fornecimento de energia geralmente são identificados quando o dispositivo deixa de funcionar como deveria. Se esses problemas não forem prontamente eliminados, poderão causar prejuízos experimentais (ruídos diversos, redução da vida útil, ocorrência de falhas, erros sistemáticos de medidas -ocasionando baixa precisão e acurácia), comprometendo a capacidade produtiva das atividades de Pesquisa e Desenvolvimento dos instrumentos.

Conhecer a natureza dos distúrbios que podem afetar a aquisição de dados em um laboratório ou experimento é essencial para definir qual atitude deve ser tomada para correção do problema. Fenômenos dependentes da temperatura costumam ser lentos e necessitam de uma análise contínua e de longos períodos de medição. Eventos espúrios e intermitentes podem demandar registros contínuos a fim de serem detectados.

No ambiente de pesquisa, principalmente nas áreas experimentais, a coleta de dados é uma etapa de extrema importância. Nesse sentido, o monitoramento adequado do ambiente, e do próprio experimento, se apresenta como uma forma de aumentar a confiabilidade e a qualidade dos dados obtidos.

1.3 Objetivo

O objetivo dessa dissertação é desenvolver uma plataforma, envolvendo aquisição, transmissão, armazenamento e visualização dos dados, que possa ser utilizada no Centro Brasileiro de Pesquisas Físicas (CBPF) ou em outros centros de pesquisa e universidades, como um sistema de monitoramento e instrumentação IoT, para ser aplicado em laboratórios e experimentos. Além disso, a segurança da informação e do sistema devem ser robustos, com a utilização de criptografia na comunicação e no *firmware*. Com isso, é esperado que o projeto seja utilizado para a medição de grandezas elétricas e variáveis ambientais, além de permitir adaptações para a medição das mais diversas grandezas.

1.4 Organização da Dissertação

Esta dissertação está organizada em 6 capítulos, contando ainda com 2 apêndices que fornecem mais detalhes sobre a implementação do *middleware* e do *firmware*.

No Capítulo 2 é feita a revisão bibliográfica sobre o tema da dissertação.

No Capítulo 3 são apresentadas as considerações utilizadas para o desenvolvimento da Plataforma IoT, sobre a escolha dos materiais utilizados, bem como a forma pela qual o projeto foi desenvolvido.

O Capítulo 4 contém os aspectos de segurança que foram considerados para o projeto.

No Capítulo 5 são apresentados os resultados obtidos com a implementação do sistema.

Por fim, o Capítulo 6 apresenta as conclusões sobre o trabalho e as perspectivas futuras, com o intuito de dar continuidade ao projeto.

Os Apêndices A e B, apresentam, respectivamente, os diagramas de classe para as bibliotecas desenvolvidas e a descrição das rotas implementadas na API para a comunicação com o *middleware*. No Apêndice C é apresentado as publicações que foram produzidas com base nos trabalhos desenvolvidos nessa dissertação.

Capítulo 2

Revisão Bibliográfica

Antes de se iniciar o desenvolvimento do sistema, foi feita uma análise do que tem sido publicado referente aos temas de arquitetura, monitoramento e medição utilizando instrumentação IoT.

2.1 Padronização da Arquitetura IoT

Com a utilização cada vez maior de arquiteturas baseadas em IoT, surgiu a necessidade de se definir as características desse tipo de ambiente, com a adoção de uma abordagem unificada para reduzir a fragmentação e permitir a utilização pela indústria. A norma IEEE 2413-2019 [1] veio para definir uma padronização que facilite o desenvolvimento de novas soluções IoT, trazendo mais segurança e facilidade para os projetistas, clientes e fornecedores.

Nessa norma, o conceito de *thing* é definido como um componente IoT que possui função, propriedade e meios de trocar informações, de forma segura, com aplicações e serviços disponíveis na nuvem (Figura 2.1).

É importante notar a relevância da camada de segurança, onde todos os sistemas devem possuir meios seguros de executar as suas funções, propriedades e comunicação. Segundo esse padrão, um sistema IoT deve possuir as seguintes características:

- Modularidade: Capacidade de se conectar a outros dispositivos;
- Agilidade: Capacidade de rapidamente ser modificado (adição/remoção de componentes ou sensores, por exemplo.);
- Capacidade de compartilhamento: Os componentes podem ser compartilhados entre outros sistemas;
- Confiabilidade: O sistema deve possuir formas de garantir a confiabilidade dos dados coletados;

- Resiliência: O sistema deve ser capaz de continuar operando após o reestabelecimento da comunicação e/ou energia após uma falha;
- Garantia de que o sistema irá funcionar de forma adequada: O sistema deve se comportar de forma adequada sob quaisquer condições de operação para o qual ele foi projetado.

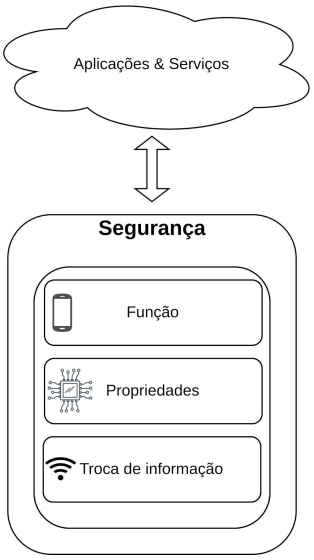


Figura 2.1: Definição de "coisa" segundo a IEEE 2413-2019 [1].

Esta norma é utilizada como referência neste trabalho para padronização do vocabulário e definição dos conceitos envolvidos.

2.2 Aspectos Sobre o Paradigma da Internet das Coisas

A complexidade do ambiente IoT está, de forma paradoxal, relacionada com a sua maior vantagem: a flexibilidade [7]. Essa liberdade possui correlação com o paradigma da Computação Ubíqua e faz com que seja árdua a tarefa de projetar sistemas IoT, pois existe uma miríade de recursos disponíveis que podem ser integrados e, até então, não existia uma padronização para isso. No artigo também fica claro a importância do desenvolvimento do *middleware*, pois é através dessa abordagem que boa parte dos problemas relacionados com a heterogeneidade de dispositivos e padrões envolvidos podem ser minimizados.

2.3 Medidores Inteligentes de Eletricidade

O artigo publicado na revista *IEEE Transactions on Industrial Informatics* [15], menciona a crescente preocupação com a qualidade de energia e a facilidade que a penetração da tecnologia IoT pode trazer nesse sentido.

O artigo define que "medidor inteligente" se refere a um sistema capaz de medir a eletricidade utilizada e/ou gerada, além de possuir a capacidade de controlar remotamente o suprimento e o corte de energia quando necessário.

O que se espera de um medidor de energia é que ele possua as seguintes características:

- Capacidade de monitoramento em tempo real ou próximo ao tempo real;
- Possibilidade de leitura dos dados de forma remota ou local;
- Controle remoto;
- Capacidade de integração com outros sistemas;
- Capacidade de captura de dados sobre a condição do sistema e a qualidade da energia(incluindo a tensão);
- Interoperabilidade com outros sistemas.

No que tange à medição elétrica, o projeto foi desenvolvido seguindo as premissas delineadas pelo que o artigo define como "estado da arte" em medição inteligente. Foram utilizadas ferramentas não proprietárias, que permitem a integração com as mais diversas tecnologias existentes.

2.4 Utilização de IoT na Condução de Experimentos

A utilização da IoT foi impulsionada nos últimos dois anos devido aos impactos causados pela pandemia, levando a um aumento nas buscas por soluções para controle e monitoramento de experimentos de forma remota.

No trabalho analisado [16] o autor cita as dificuldades para a condução de pesquisas no período de isolamento e desenvolve um aparato que permite o monitoramento do laboratório, através de câmeras, utilizando IoT e permitindo a condução dos experimentos de forma remota.

2.5 Utilização de IoT no Gerenciamento de Laboratórios

Existe um nicho de mercado que vem ganhando destaque, o de *smart labs* [17] ou *Internet of Laboratory Things*(IoLT), e já existem empresas especializadas em instrumentação IoT para laboratórios, como a *TetraScience* [18].

Com o monitoramento dos equipamentos é possível ampliar a gama de dados coletados, permitindo um melhor controle da utilização dos ativos. Também é possível identificar os usuários que estão fazendo uso, o horário e o tempo em que os equipamentos estão em operação.

Outra possibilidade que surge com o monitoramento é o da utilização dos dados obtidos para identificar padrões que possam indicar a necessidade de manutenção (manutenção preditiva) [19]. Com isso é possível programar os períodos de parada de forma a minimizar as ocorrências de falhas durante as atividades de pesquisa.

2.6 A Evolução Proporcionada pela Internet das Coisas

De acordo com o artigo *Hype Cycle for Internet of Things*, de 2019, publicado na Gartner [20], estamos na subida da curva de desenvolvimento e expectativas com relação as tecnologias que são trazidas com o advento da Indústria 4.0.

Algumas áreas que em breve irão se destacar envolvem Cibersegurança, *Blockchain*, Setor Financeiro e plataformas que utilizam IoT. A expectativa é que esse *boom* se torne realidade por volta de 2029. Nesta etapa final as tecnologias já estarão maduras e iremos conviver com elas de forma natural.

2.7 Aspectos de Segurança Para um Dispositivo IoT Utilizando o ESP32

Em [21], Michal faz um apanhado bastante interessante sobre os aspectos de segurança e as vulnerabilidades que podem ser encontradas em sistemas IoT. Os focos são em possíveis fragilidades de *software* e *hardware* que podem ser encontradas nos dispositivos que utilizam o ESP32.

2.8 Plataforma IoT

A definição de uma plataforma para o projeto levou em consideração algumas premissas focadas na resolução dos problemas vivenciados nos ambientes de pesquisa. Nas medições que envolvem os estudos de compatibilidade eletromagnética, por exemplo, o interesse maior são nos valores do campo elétrico e campo magnético dentro de um raio determinado, para o caso de linhas de transmissão [22]. Valores altos de umidade relativa do ar podem permitir a ocorrência do Efeito Corona, levando a perdas na transmissão, impactando no fluxo máximo de potência que pode ser transmitida. As características da rede elétrica associadas às condições climáticas desfavoráveis podem impactar na operação de equipamentos.

Devido às especificidades que cada experimento possui, foi pensado qual seria um conjunto mínimo de sensores que pudessem ser utilizados para permitir a escalabilidade de forma adequada, com fácil integração e impacto mínimo em um sistema que já esteja em operação.

Com isso vem o conceito de **Arquitetura de Referência** [23] [24]. A arquitetura deve definir um conjunto inicial de blocos de construção que levem em consideração os requisitos dos ambientes onde a tecnologia será aplicada. Essa arquitetura de referência possui natureza abstrata e deve envolver o conhecimento e experiências dentro de um domínio de aplicação, com o objetivo de orientar o desenvolvimento, a padronização e a evolução de sistemas de *software* [7] [25] [26].

2.9 Requisitos da Plataforma

Uma plataforma IoT deve possuir alguns requisitos considerados fundamentais [27], que podem ser definidos como:

- **Interoperabilidade**: a plataforma de *middleware* possui extrema importância nesse cenário, pois ela será a responsável por gerenciar a comunicação entre os dispositivos e as aplicações;
- **Descoberta e gerenciamento de dispositivos**: o sistema deve ser capaz de reconhecer quando um dispositivo foi conectado, além de possibilitar o controle de forma remota;
- **Interface de alto nível**: o sistema deve ser capaz de permitir o acesso de forma que seja facilmente interpretado pelos seres humanos. Por exemplo, um ambiente onde as variáveis monitoradas possam ser visualizadas graficamente.
- **Ciência de contexto**: o *software* deve ser capaz de coletar, gerenciar e processar informações sobre o contexto ao qual o dispositivo está em operação,

como a localização e informações da rede, por exemplo;

- **Escalabilidade**: a escalabilidade envolve a capacidade do sistema de ser capaz de assimilar um número crescente de dispositivos conectados e continuar funcionando corretamente;

- **Gerenciamento de grandes volumes de dados**: o *middleware* deve ser capaz de suprir a demanda crescente de tráfego de informação que será causada pela quantidade cada vez maior de dispositivos em operação. Nesse sentido, a estrutura da rede, processamento e armazenamento da informação devem ser cuidadosamente planejados;

- **Segurança**: a plataforma deve ser capaz de manter a integridade e a privacidade dos dados coletados, assim como proteger os dispositivos e os recursos disponibilizados na rede;

- **Adaptação dinâmica**: a plataforma de *middleware* deve ser executada em um ambiente de alta disponibilidade, sendo capaz de reagir à mudanças no contexto em que ela e os dispositivos conectados a ela estejam inseridos.

Quanto mais um projeto estiver adequado a esses requisitos, maior será a facilidade de adaptação para novos recursos e tecnologias que vierem a surgir.

Capítulo 3

Materiais e Métodos

Neste capítulo são descritos o modelo de arquitetura IoT que foi implementado no projeto, os materiais utilizados para montagem do equipamento bem como os testes que foram realizados.

3.1 Arquitetura do Projeto

Este projeto utilizou como arquitetura a abordagem de *Cloud of Things*(CoT) [28]. A arquitetura IoT utilizada nesse projeto foi a de três camadas. A arquitetura CoT tem como finalidade prover a integração de recursos heterogêneos de objetos físicos em uma rede com computação em nuvem, através de uma abordagem em camadas. Nesse projeto, foram, implementadas as Camadas de Nuvem, Névoa e de Dispositivos.

O projeto foi dividido em sete etapas:

- Etapa 1 : Análise dos requisitos;
- Etapa 2 : Escolha do material;
- Etapa 3 : Montagem do equipamento;
- Etapa 4 : Implementação da arquitetura de comunicação;
- Etapa 5 : Desenvolvimento do *Middleware*;
- Etapa 6 : Visualização dos dados e desenvolvimento do Supervisor;
- Etapa 7 : Integração e testes.

Para o desenvolvimento do sistema, foram levadas em consideração as seguintes premissas:

- Forma de aquisição dos dados de interesse;
- Facilidade na aquisição dos componentes;
- Utilização de ferramentas de código aberto;
- Facilidade para modularização;
- Facilidade para adição de novos sensores;
- Robustez para aplicação em diversos ambientes.

3.2 Análise dos requisitos

Nesta etapa foram determinadas as necessidades e expectativas para a plataforma desenvolvida.

O processo de análise de requisitos envolveu as seguintes etapas [29]:

Etapa 1: Reconhecimento do problema.
Identificar quais são partes interessadas (*stakeholders*), que são os responsáveis por determinar o que será incluído no escopo do projeto. Em seguida, foram identificados os usuários finais da plataforma e quais eram as expectativas para o projeto. Nesse projeto os *stakeholders* são os chefes de laboratórios e diretores das instituições. Os usuários são os alunos e técnicos que utilizam os equipamentos e executam os experimentos.

Etapa 2: Avaliação do problema e a definição da solução.
Definição da solução com base em pesquisas na literatura e expectativas e necessidades levantadas com os usuários. Construção de protótipos.

Etapa 3: Especificação dos requisitos.
Consolidação das funções necessárias, formas de visualização, métodos de aquisição de dados, levantamentos dos componentes, custo do projeto e formas de aquisição do material necessário.

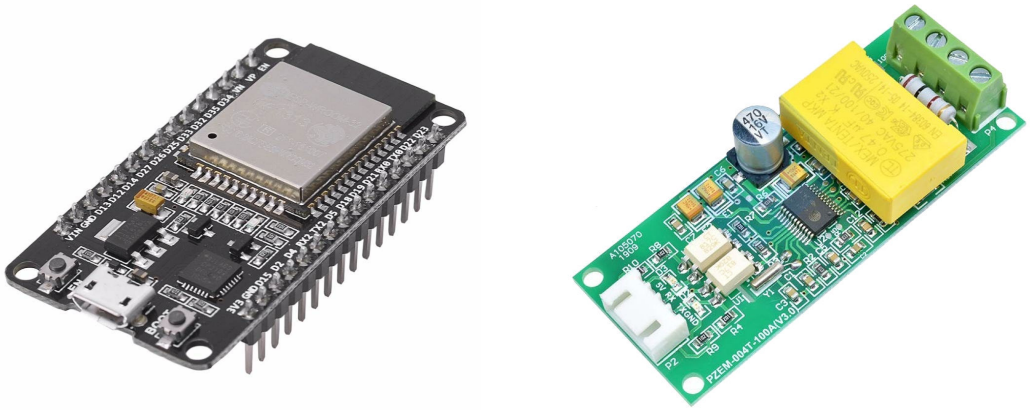
Etapa 4: Interpretação e registro dos requisitos.
Determinação de quais requisitos são realmente alcançáveis e elaboração da documentação de cada um deles. Definição dos requisitos do projeto, do produto, funcionais e não funcionais. Nesta etapa também foi feita a priorização dos requisitos, análise de impacto e resolução de conflitos.

Etapa 5: Revisão.
Nesta fase foram analisadas as possíveis redundâncias, inconsistências e omissões do sistema.

3.3 Escolha do Material

Após a etapa de análise de requisitos, seguindo as premissas que caracterizam um sistema IoT, com base na literatura e utilizando a experiência adquirida com o projeto [30] -desenvolvido com base nesse trabalho - foi feita a escolha dos seguintes componentes:

- Microcontrolador ESP32 (Figura 3.1a);
- Módulo PZEM-004T (Figura 3.1b), para medição de energia;
- Transformador de corrente não invasivo PZCT-02 (Figura 3.2a);
- Sensor de pressão, temperatura e umidade BME-280 (Figura 3.2b);
- Display LCD 20x4 com módulo de comunicação I2C.



(a) ESP32. (b) PZEM-004T

Figura 3.1: Microcontrolador e medidor de energia.

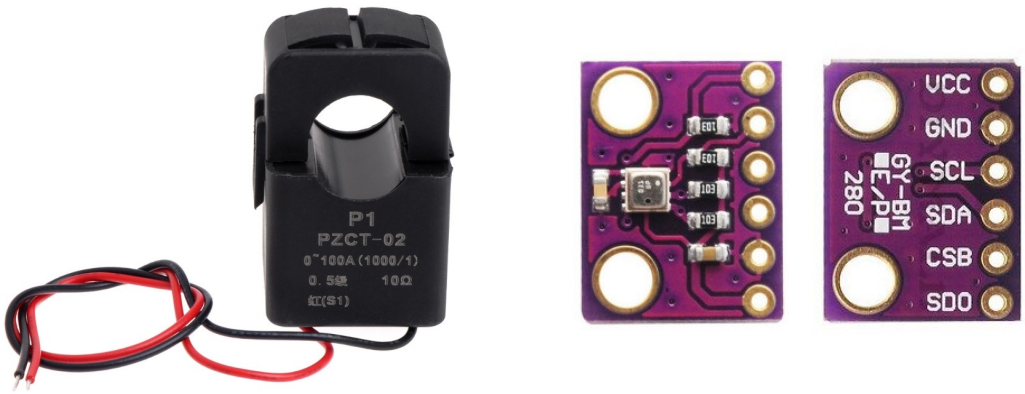
O ESP32 [31] faz parte de uma série de microcontroladores de baixo consumo de energia com Wi-Fi e Bluetooth integrados. O *hardware* é criado e desenvolvido pela empresa Espressif Systems e é fabricado pela TSMC [32], usando o processo litografia de 40 nm (Figura 3.1a). O dispositivo ainda conta com com uma boa capacidade de processamento devido ao seu processador - que possui 240 MHz, dois núcleos e aceita instruções de 32 bits.

O PZEM-004T (Figura 3.1b) é um módulo baseado no *SOC* SD3004, voltado para monitoramento elétrico. O SD3004 calcula o valor RMS para tensão e corrente, além da potência ativa, fator de potência e frequência da rede. Ele pode ser utilizado para medição das grandezas de energia, tensão e corrente, além de possuir uma interface aberta e documentada, de fácil comunicação e integração. Além de medir

a tensão, corrente e a frequência da rede, ele também calcula o fator de potência e a potência ativa (Tabela 3.1).

PZEM-004T	
Tensão de Alimentação [V]	3,3 a 5,5
Tensão [V]	80 a 260 ± 0,1 @0,5%
Corrente [A]	0 a 100 ± 0,001 @0,5%
Potência ativa [W]	0 a 23kW ± 0,1 @0,5%
Fator de potência [Φ]	0,00 a 1,00 ± 0,01 @1%
Frequência [Hz]	45 a 65 ± 0,1 @0,5%
Energia ativa [kWh]	0 a 9999,00 ± 0,001 @0,5%
Interface	RS485(Modbus)

Tabela 3.1: Dados do sensor PZEM-004T [2].



(a) Pzct. (b) BME280

Figura 3.2: Transformador de corrente e sensor de pressão, umidade e temperatura.

O PZCT-02 é um mini transformador de corrente no formato de alicate. É pequeno, fácil de transportar e possui relação de 1000:1 (entrada de até 100 A e saída de até 100 mA). Ainda possui boas propriedades mecânicas, resistência ao ambiente e alta capacidade de isolamento de tensão (Figura 3.2a).

O BME280 [4] é um sensor que vem no módulo GY-BME280 (Figura 3.2b) e tem como característica a capacidade de medir pressão, temperatura e umidade, além de funcionar com interfaces de comunicação I2C ou SPI. Possui tensão de operação de 3,3V e pode ser utilizado com o ESP32 (Tabela 3.3).

Para a maioria dos projetos desenvolvidos dentro das condições de operação em que o equipamento irá encontrar, as características de precisão e confiabilidade

Características de operação do PZCT-02	
Temperatura [^oC]	-40 a 85
Umidade [%]	< 90 @40^oC
Frequência da rede [Hz]	50 a 60
Classe	B @130^oC
Isolamento [Ω]	1000M
Resistência Interna [Ω]	10
Faixa de medição (entrada) [A]	0 a 100
Faixa de saída [mA]	0 a 100

Tabela 3.2: Dados do PZCT-02 [3].

BME280	
Tensão de alimentação [V]	3,3
Pressão [hPa]	300 a 1100 ± 1,00
Temperatura [^oC]	-40 a 65 ± 0,5
Umidade [%RH]	0 a 100 ± 1
Interfaces	I2C e SPI

Tabela 3.3: Dados do módulo com o BME280 [4].

dos equipamentos são suficientes. A utilização de sensores mais robustos e precisos encarecem o projeto e deve ser avaliada de acordo com a necessidade.

3.4 Montagem do Equipamento

Como o equipamento tem por objetivo ser aplicado em ambientes de pesquisa, ele foi implementado com a ideia de ser flexível e minimamente invasivo. A parte de conexão ao barramento de alimentação deverá ser feito por pessoal habilitado, pois muitas vezes será necessário fazer a ligação sem interromper a operação do laboratório.

O diagrama representativo da montagem e ligações pode ser visto na Figura 3.3. A conexão do módulo de medição é feita após o disjuntor principal do circuito, no ponto de alimentação das cargas. Os transformadores de corrente não-invasivos (TC's) são conectados aos cabos na saída do disjuntor (DJ-1). Cada módulo de medição elétrica (PZEM) é conectado a fase de referência e o neutro. Essa conexão pode ser feita com a utilização de conectores com isolamento. Por convenção, a alimentação do módulo é feita através das fases A e C.

Uma das ideias do sistema é que ele seja flexível, permitindo a sua adaptação para diversos tipos de configurações. Para alcançar tal objetivo, foram desenvolvidas duas classes em C++ para facilitar os projetos: uma para a comunicação com o medidor de energia e outra para integração de diversos modelos de sensores.

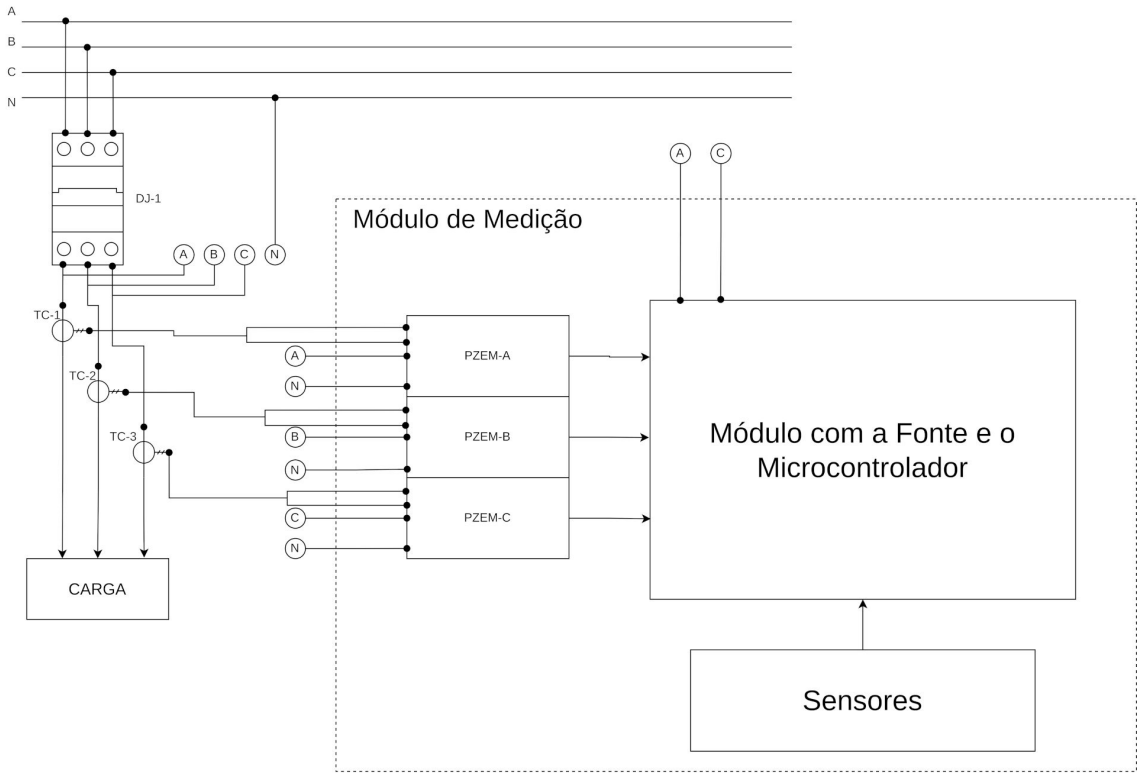


Figura 3.3: Diagrama de conexão do sistema.

A primeira classe foi desenvolvida para a comunicação do ESP32 com o sensor PZEM-004T (Figura 3.1b). Durante o desenvolvimento da dissertação foi encontrada uma biblioteca com a qual foi possível estabelecer a comunicação com apenas dois medidores, utilizando o ESP32 [33]. A alternativa encontrada foi adaptar o código existente para o ESP8266, que utiliza comunicação serial, permitindo que o ESP32 fosse capaz de se comunicar com o sensor através do protocolo Modbus (RS-485), visto que o módulo de medição de energia é compatível e possui documentação disponível [2]. O ESP8266 é o modelo de microcontrolador predecessor do ESP32. O diagrama da classe desenvolvida pode ser vista na Figura A.1 (Apêndice A).

Após essa adaptação o procedimento para conexão dos sensores para medição elétrica passou a envolver duas etapas:

- Etapa 1: Conexão de cada sensor ao equipamento (microcontrolador) e definição do endereço que o sensor terá no barramento.
- Etapa 2: Conexão de todos os sensores ao barramento de comunicação e requisição dos dados.

Com o desenvolvimento dessa classe, foi possível fazer a comunicação com mais

de dois sensores conectados ao mesmo barramento I2C.

Para evitar a necessidade de ter que reprogramar o projeto toda vez que for utilizar um sensor novo, foi desenvolvida uma classe para a integração das funções utilizadas por diversos sensores. Desta forma tornou-se mais simples o desenvolvimento de novos projetos. O diagrama da classe desenvolvida pode ser vista na Figura A.2 (Apêndice A)

O *firmware* foi desenvolvido de forma modular, o que permite adaptar o projeto de acordo com as necessidades. Este código pode ser compilado para que o sistema funcione em três modos de operação, definidos no projeto como:

- modo de operação autônomo (1);
- modo de operação por mensagem (2);
- modo de operação de monitoramento (3).

Estes modos são selecionados através de definições no código fonte.

No **modo de operação autônomo** o *firmware* executa o seu próprio servidor web, podendo ser acessado através do endereço IP. Nessa configuração é possível operar o dispositivo de forma independente, permitindo a atuação e monitoramento através de uma interface limitada (Figura 3.4). Uma das utilizações do **modo de operação autônomo** é quando a intenção for apenas exibir o que está sendo medido, algo útil para operar em paralelo com outros sistemas, em aplicações residenciais ou ambiente com monitoramento local.

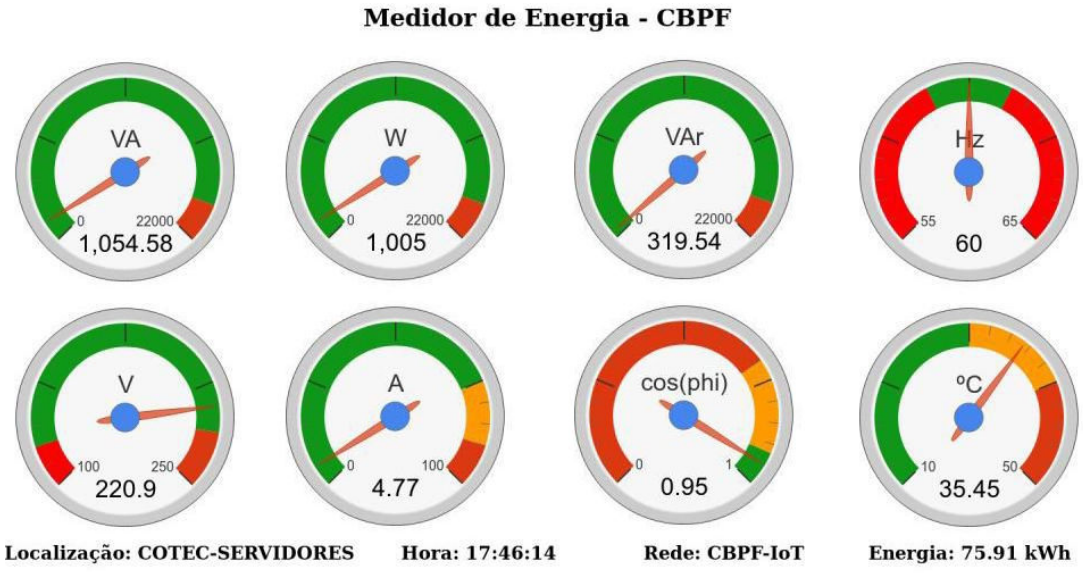


Figura 3.4: Exemplo do modo de operação autônoma do equipamento em teste na sala de servidores da COTEC do CBPF.

No **modo de operação por mensagem** o equipamento trabalha na configuração *publish/subscriber*, utilizando rotas. Uma rota nada mais é que um endereço web que será processado e retornará uma resposta pelo servidor. Através de uma interface é possível requisitar os estados dos sensores ou enviar comandos para atuação. Nesse modo o dispositivo funciona de forma passiva, semelhante ao modo autônomo, com a diferença de que para cada requisição feita, através de um caminho específico (http://10.0.0.5/medidas/, por exemplo), ele irá retornar uma resposta no formato JSON.

No **modo de operação de monitoramento**, o sistema coleta os dados dos sensores e os envia para um banco de dados através de um método de acordo com a API desenvolvida. Nesse modo de operação o equipamento funciona de forma ativa, fazendo a aquisição e o envio dos dados de forma independente, sem a necessidade de interferência externa. Ao ser ligado o equipamento se conecta a rede Wi-Fi e procura o servidor para o envio dos dados. O fluxograma de operação desse modo pode ser visto na Figura 3.5.

O desenvolvimento de cada um desses modos de operação foi feito de forma independente durante a implementação do sistema, para atender a necessidades específicas de utilização que foram surgindo e, ao final, foram incorporados ao projeto.

A partir dos três modos desenvolvidos, a forma de operação escolhida para ser utilizada no projeto foi o modo 3 (modo de operação de monitoramento).

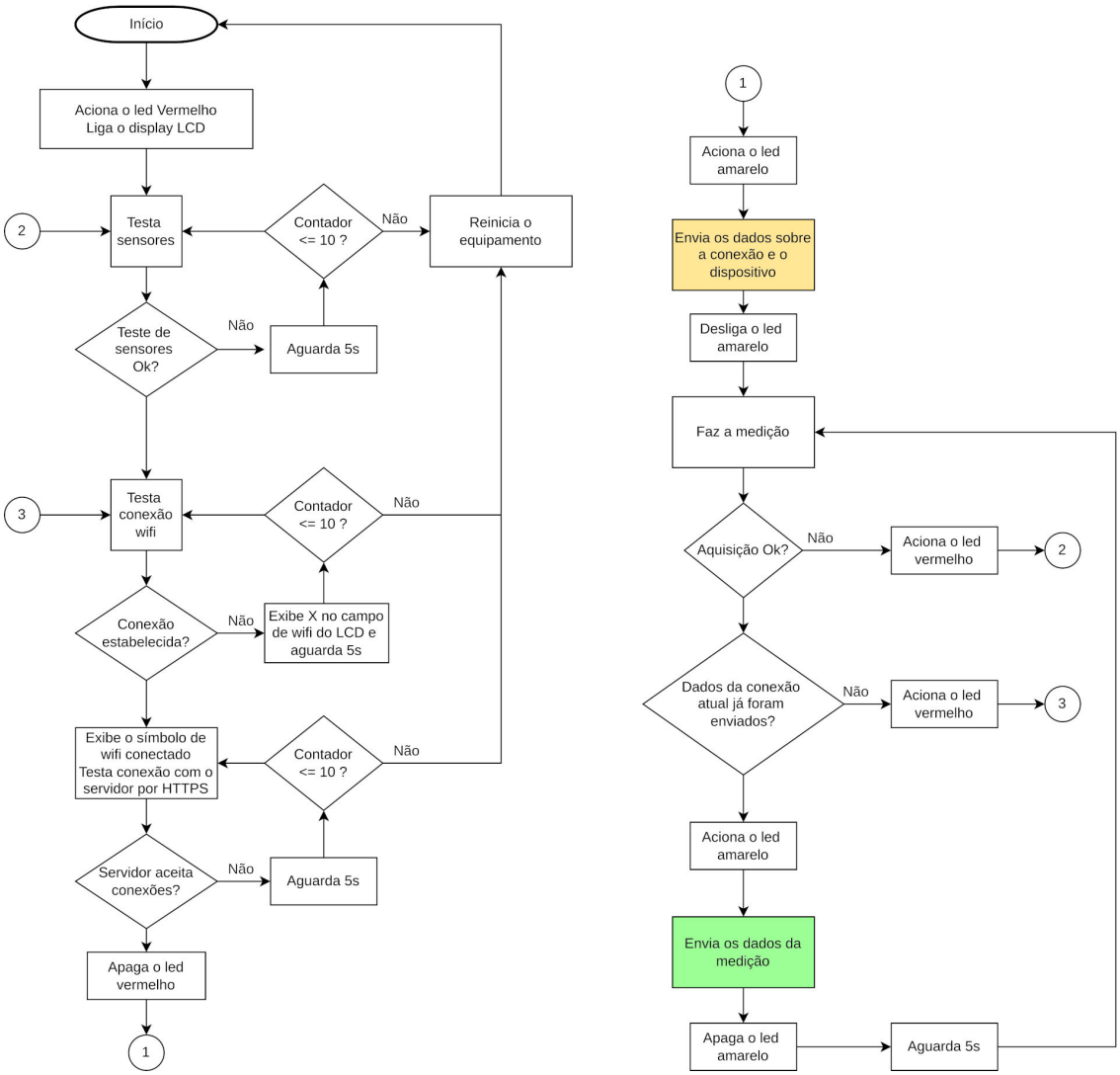


Figura 3.5: Fluxograma de operação do *firmware* no modo de monitoramento.

Na etapa destacada em amarelo na Figura 3.5 o equipamento faz o envio de seis atributos, através do método POST na rota IP/medidores/. Essa é a etapa de identificação, onde o equipamento informa para o sistema qual é o seu endereço na rede, identificação e localização, além de outras informações relevantes sobre o dispositivo, como pode ser visto na Tabela 3.4.

Já na etapa destacada em verde na Figura 3.5 o equipamento faz o envio de 26 atributos, através do método POST na rota IP/medidas/ (Tabela 3.5).

Variável	Descrição
local	Localização do sensor
ip	Endereço IP que o equipamento possui na rede
mac	Endereço MAC do equipamento
rede	Nome da rede Wi-Fi na qual o equipamento se conectou
versao	Versão do *Firmware*
compilado	Data de compilação do *Firmware*

Tabela 3.4: Tabela com as informações enviadas através da rota "medidores".

Variável	Descrição
local	Localização do sensor
sensor_name	Nome atribuído ao equipamento
ip	Endereço IP que o equipamento possui na rede
mac	Endereço MAC do equipamento
rede	Nome da rede Wi-Fi a qual o equipamento se conectou
temperatura	Temperatura
umidade	Umidade relativa do ar
pressao	Pressão atmosférica
tensao_a	Tensão na fase A
tensao_b	Tensão na fase B
tensao_c	Tensão na fase C
corrente_a	Corrente na fase A
corrente_b	Corrente na fase B
corrente_c	Corrente na fase C
potencia_a	Potência na fase A
potencia_b	Potência na fase B
potencia_c	Potência na fase C
energia_a	Energia consumida na fase A
energia_b	Energia consumida na fase B
energia_c	Energia consumida na fase C
frequencia_a	Frequência da rede elétrica na fase A
frequencia_b	Frequência da rede elétrica na fase B
frequencia_c	Frequência da rede elétrica na fase C
fator_potencia_a	Fator de potência na fase A
fator_potencia_b	Fator de potência na fase A
fator_potencia_c	Fator de potência na fase A

Tabela 3.5: Tabela com as informações enviadas através da rota "medidas".

A proposta é que quando seja necessário o monitoramento elétrico, o *hardware* de medição seja conectado, de forma não invasiva, entre o ponto de alimentação e o equipamento a ser monitorado, como pode ser visto na Figura 3.6.

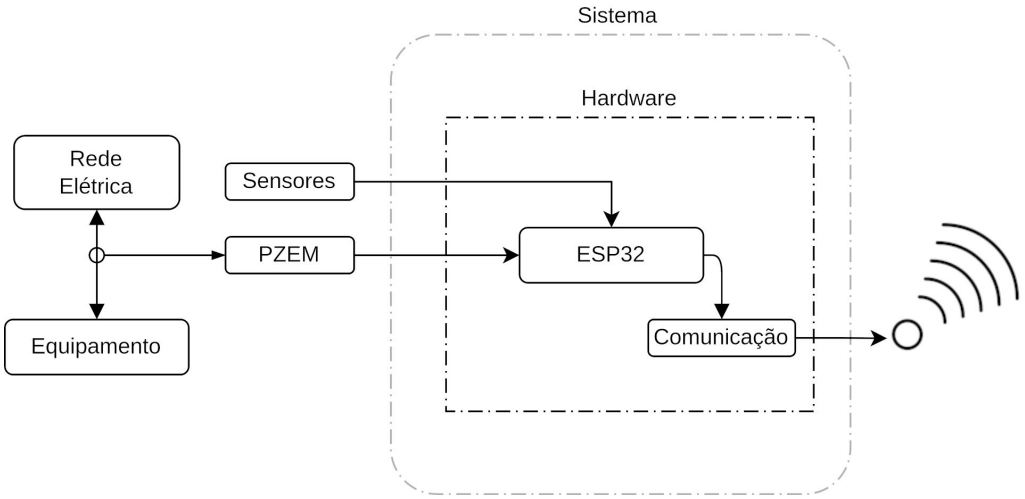


Figura 3.6: Diagrama de conexão do dispositivo desenvolvido.

Todo o equipamento do protótipo foi acomodado dentro de uma caixa, modelo Patola PB-114 [34]. Também foi desenvolvido e impresso, utilizando uma impressora 3D, um suporte para acomodação dos medidores dentro da caixa (Figura 3.7).

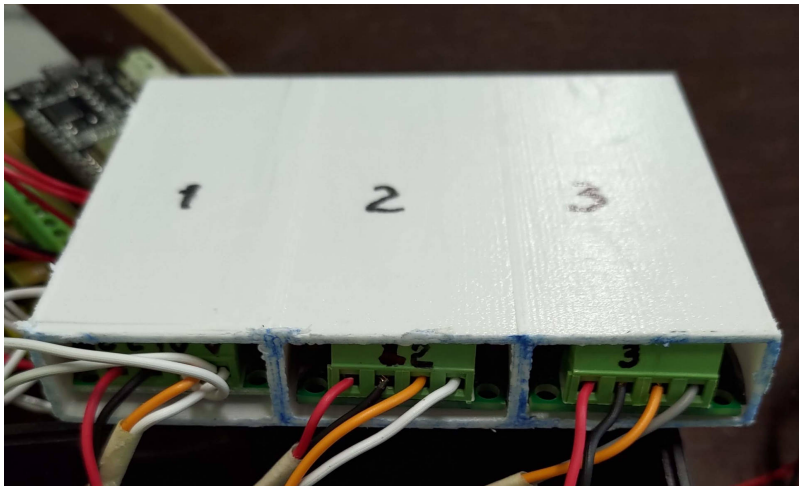


Figura 3.7: Suporte feito utilizando a impressora 3D para acomodação dos medidores de energia.

Para visualização local, o equipamento possui um visor onde são exibidas algumas informações úteis, como: dados de temperatura, umidade e pressão atmosférica, informações sobre a conexão do equipamento à rede Wi-Fi, tensão, corrente e fator de potência por fase, como pode ser visto na Figura 3.8.

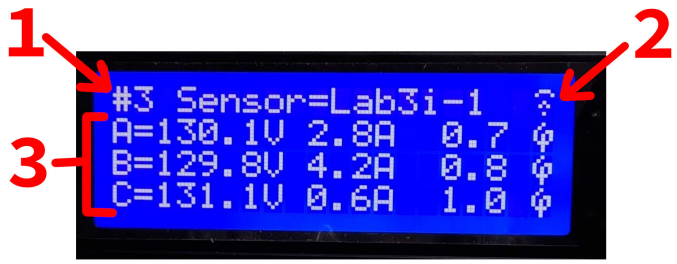


Figura 3.8: Painel do equipamento com algumas das informações coletadas.

O painel (Figura 3.8) é dividido em três regiões onde as informações estão dispostas.

- Região 1: São apresentadas, em páginas numeradas de #1 a #5, as informações sobre a temperatura, umidade, pressão atmosférica, ID do dispositivo, rede Wi-Fi a qual ele está conectado e endereço IP.
- Região 2: Apresenta o estado da conexão Wi-Fi. O símbolo fica com um X caso a conexão seja perdida.
- Região 3: São exibidos os dados de tensão, corrente e fator de potência para cada fase.

Para avaliação da viabilidade econômica de implantação do equipamento está descrito na Tabela 3.6 uma estimativa de custo do *hardware* do projeto utilizando valores típicos encontrados em sites de venda pela Internet, como o Amazon. Esse custo é referente ao protótipo, com valores atualizados em maio de 2022. É necessário considerar que ainda existem os custos de infraestrutura para instalação do equipamento e desenvolvimento e adaptação do *software*.

Estimativa de custo do projeto		
Equipamento	**Custo Unitário**	**Custo Total**
ESP32	US$ 8,00	US$ 8,00
Módulo PZEM	US$ 15,00	US$ 45,00
PZCT-02	US$ 10,00	US$ 30,00
BME280	US$ 7,00	US$ 7,00
Display LCD 20x4	US$ 5,00	US$ 5,00
Cabos, conectores e acessórios	US$ 30,00	US$ 30,00
Total		US$ 125,00

Tabela 3.6: Estimativa de custo para o sistema de monitoramento para uma rede trifásica.

3.5 Implementação da Arquitetura de Comunicação

Dentro do CBPF foi possível implementar, desde o início, todas as etapas da arquitetura de comunicação. O sistema de nuvem, envolvendo o ambiente de virtualização VMWare e o servidor de armazenamento de dados, foram instalados e configurados em um *cluster*. Esse *cluster* é composto de 3 servidores com a seguinte configuração cada: dois processadores, 40 *threads*, 256 GB de memória ram, 1 TB para o sistema operacional e 4 TB de armazenamento local. O ambiente de armazenamento (Storage) possui dois blocos em Raid 6 com 160 TB de espaço disponível.

Nesta etapa a topologia do ambiente (Figura 3.9) envolveu duas camadas, conforme definido no Capítulo 3. O servidor, que está na camada de nuvem, foi implementado em uma máquina virtual utilizando a distribuição Rocky Linux. Já os equipamentos estão na camada de dispositivos (ESP32) e enviam os dados de forma criptografada para o servidor.

A comunicação entre o módulo desenvolvido - representado na Figura 3.9 pelo ESP32 - e o servidor (VM Server), é estabelecida através de conexão Wi-Fi, com autenticação na rede baseada na especificação WPA2-PSK, utilizando criptografia TLS v1.2 e autenticação para o envio através da API por chaves de autenticação.

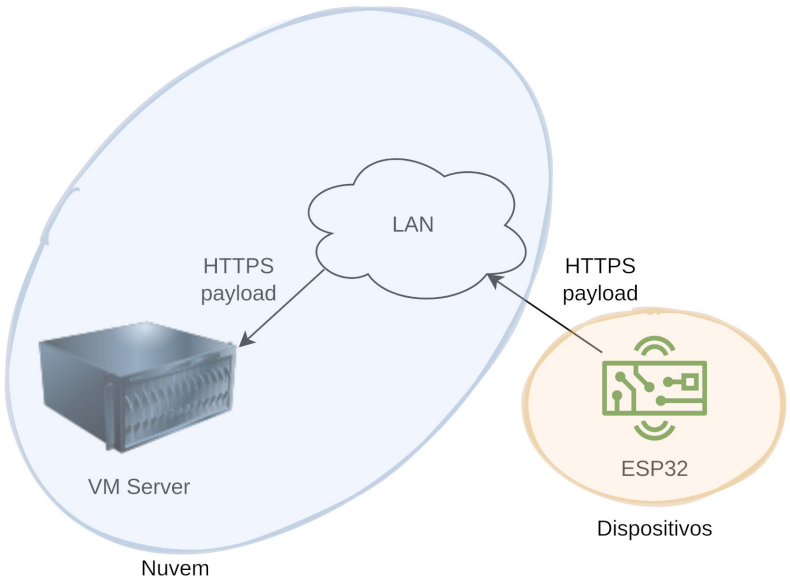


Figura 3.9: Topologia da comunicação.

A segurança é feita através de um *firewall*, sistema de detecção de intrusão (IDS) e a utilização do *SELinux*.

O *SELinux* foi utilizado por ser uma arquitetura de segurança desenvolvida pela *National Security Agency* (NSA) do governo dos Estados Unidos e que define níveis de controle de acesso para aplicações, processos e arquivos em um sistema [35]. Essa característica é útil para conter o escalonamento de privilégios, devido a exploração de falhas que ainda não foram publicadas ou que ainda não possuem correções.

Também foram implementadas políticas de *backup*, dentro do sistema de virtualização, para recuperação dos dados em caso de desastres. A comunicação com a API é feita de forma monitorada por um *Web Application Firewall* (WAF). Para implementar uma configuração minimamente segura no servidor web, foram utilizadas como ponto de partida as recomendações do *Mozilla SSL Configuration Generator* [36].

A arquitetura final do fluxo dos dados pode ser vista na Figura 3.10. O protocolo de criptografia escolhido foi o TLS v1.2, por ser amplamente suportado pelos sistemas e por existirem bibliotecas disponíveis para o ESP32, além de ser suficientemente seguro para a aplicação.

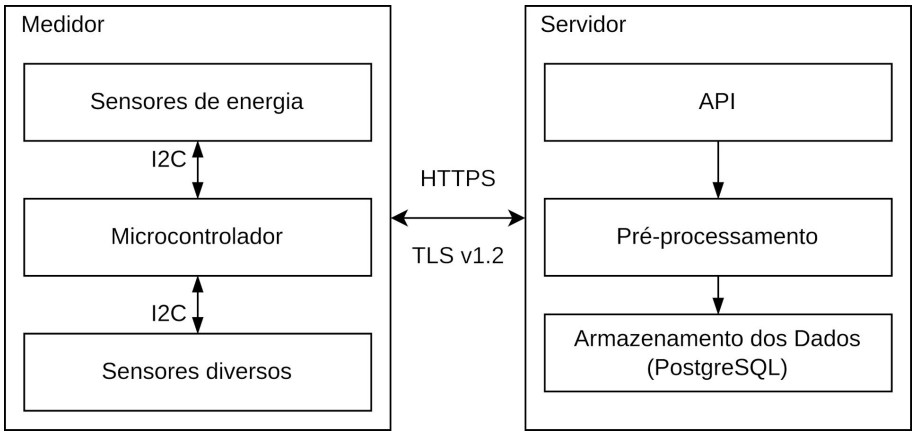


Figura 3.10: Detalhe da arquitetura.

3.6 Desenvolvimento do Middleware

Middleware é um software que permite um ou mais tipos de comunicação entre aplicativos ou componentes de aplicativos em uma rede distribuída. O objetivo dessa abordagem de implementação é facilitar a comunicação entre dispositivos que não foram projetados para se comunicarem uns com os outros, possibilitando a integração de maneira inteligente [37].

O termo recebeu esse nome porque o primeiro *middleware* agia como uma interface que operava na mediação entre um *front-end* de aplicativo, ou cliente, e um recurso de *back-end*, como um banco de dados ou um dispositivo de hardware especializado do qual o cliente pode solicitar dados.

É curioso notar que o termo foi cunhado pela primeira vez em um relatório após a conferência de Engenharia de Software da OTAN de 1968. Essa conferência estava definindo o que, no futuro, viria a ser o campo da engenharia de software e reuniu profissionais para discutir temas como desenvolvimento, produção e distribuição de software [37].

O *middleware* desta dissertação foi desenvolvido dentro da filosofia de uma *Application Programming Interface* (API). Com isso a API funciona como uma camada externa, com um conjunto de ferramentas, definições e protocolos que permitem que as aplicações se comuniquem entre si, possibilitando a conexão de serviços diferentes através de uma camada em comum.

É possível exemplificar o funcionamento de uma API através do exemplo de um garçom em um restaurante. No restaurante o garçom é o responsável por interpretar o pedido do cliente (checar se o pedido está disponível no cardápio, por exemplo), fazer a requisição na cozinha e entregar o que foi pedido. De forma semelhante, uma API processa a requisição, faz a verificação se ela é válida, faz a requisição ao servidor e entrega o resultado do que foi processado pelo servidor ao cliente.

Observe que nesse modelo o processamento é transparente para o usuário ou sistema que faz a requisição. Isso possibilita a comunicação com os mais diversos tipos de equipamentos, bastando informar à API como ela deve processar cada tipo de requisição.

A API desenvolvida é responsável por receber os dados obtidos pelo ESP32 e foi elaborada em Python, utilizando a ferramenta Django [38] e o *framework* Django REST [39].

Django é umas das ferramentas para desenvolvimento *web* mais utilizadas no mundo. Ela é escrita na linguagem de programação Python, possui um ambiente de gerenciamento integrado e foi lançada em 2005, sob a licença BSD.

Esta ferramenta foi escolhida por ser baseada em Python, que é uma linguagem amplamente utilizada no meio científico. A motivação técnica para a escolha da linguagem e das bibliotecas também levou em consideração os seguintes aspectos:

- Vasta documentação disponível;
- Desenvolvimento rápido;
- Segurança;
- Alta escalabilidade;
- Amplamente testado.

A comunicação com a API é feita através da utilização de uma credencial de acesso (*API Key*) e todas as requisições são respondidas no formato *JavaScript Object Notation* (JSON). JSON é um formato leve, simples e de fácil leitura. É baseado na linguagem JavaScript e é amplamente utilizado para troca de informações entre sistemas.

A utilização da *API Key* é importante pois insere mais uma camada de segurança no sistema. Voltando ao exemplo do garçom no restaurante, é como se o processamento do pedido (requisição) só fosse feito após o fornecimento de uma palavra secreta para o garçom, caso contrário a requisição será ignorada.

A filosofia de desenvolvimento utilizada pelo Django é baseada no padrão MVT (*Model View Template*). Este padrão determina que a execução de uma tarefa deve ser baseada em uma requisição feita por um usuário e mapeada através de uma URL (rota). A implementação da API (descrita no Apêndice B) está representada no diagrama da Figura 3.11.

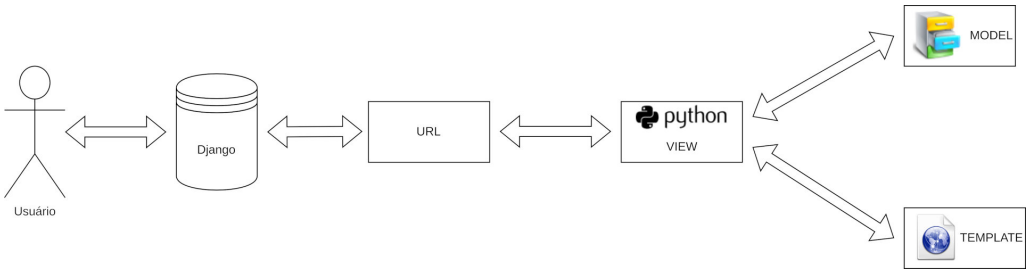


Figura 3.11: Implementação da API.

Na Figura 3.11 o Usuário faz a requisição ao sevidor Django que, em seguida, faz a chamada da *View*, através da URL requisitada. Caso essa requisição necessite de alguma informação armazenada em um banco de dados, o *Model* é processado e a resposta é enviada para a *View*. Após isso, essa informação é processada novamente pela *View* e integrada ao *Template* para, na sequência, ser devolvida para o Usuário.

O framework Django REST utiliza a filosofia REST (*REpresentational State Transfer*) que foi inicialmente sugerida por Roy Thomas Fielding em sua Tese de Doutorado [40]. O modelo é simples e possui a vantagem de operar com o protocolo HTTP, sem a necessidade do desenvolvimento de novas tecnologias ou padrões. Com isso a manipulação de dados no servidor pode ser feita através dos métodos GET, PUT, POST e DELETE, que correspondem as operações de leitura, inserção, atualização e remoção no banco de dados, por exemplo.

3.7 Desenvolvimento do Supervisor

O sistema supervisor foi desenvolvido de forma a ser integrado ao projeto da API - com isso também foi utilizada a liguagem de programação Python, além do *framework* Django. Esse ambiente tem por objetivo ser um integrador para visualizar e trabalhar com toda a informação que for coletada pelos sensores/hardware.

Dentro da IEEE 2413-2019 [1] esse sistema pode ser classificado dentro do escopo de um IOC (*Intelligent Operation Center*) para tornar mais eficiente a operação, monitoramento e controle das variáveis coletadas pelos equipamentos.

Ele tem por objetivo integrar a visualização dos dados e o gerenciamento do ambiente monitorado, permitindo a implementação de novos tipos de análises e controle.

O desenvolvimento do supervisor ainda é subdividido em duas camadas de programação: a parte de *back-end*, envolvendo a programação em Python, e a parte de *front-end*, envolvendo a utilização do JavaScript e CSS.

3.8 Integração e Testes

Nesta etapa cada sensor utilizado para criar a biblioteca (Apêndice A) foi conectado ao ESP32 e foram feitos testes de unidade do *firmware*, para certificação de que os valores obtidos estão de acordo com o esperado. Após a validação dos resultados, obtidos a partir da conexão de cada sensor de forma independente, os sensores utilizados no projeto foram conectados de forma conjunta - para avaliar possíveis problemas de integração - até que todos estivessem ligados ao ESP32. A sequência de conexão utilizada foi: PZEM 1, PZEM 2, PZEM 3 e BME280.

Ao final dos testes, os valores obtidos foram comparados com os de um multímetro da Fluke, conforme podem ser vistos na Figura 3.12.

Esse multímetro (Modelo 303) possui precisão para medição em CA de $1,5\% \pm 5$[V] e de $1,8\% \pm 5$[A]. Esses resultados confirmaram as expectativas de que os valores medidos pelo equipamento desenvolvido satisfazem às necessidades do projeto na parte de monitoramento elétrico.

Os resultados obtidos neste capítulo demonstraram a viabilidade do projeto e evidenciaram que a parte de comunicação e utilização da rede é intrínseca a um projeto em IoT. Com isso a parte de segurança não pode ser negligenciada, pois ela esta diretamente relacionada com a confiabilidade do sistema. Essa parte será discutida de forma mais detalhada no próximo capítulo.

(a) Fase A.

(b) Fase B.

(c) Fase C.

Figura 3.12: Comparação dos valores medidos pelo equipamento, exibidos por fase.

Capítulo 4

Segurança IoT

Com a crescente quantidade de equipamentos conectados à Internet, também aumentam os números relativos a ataques que exploram as vulnerabilidades desses dispositivos. Com isso, o padrão IEEE 2413-2019 [1] destacou a importância da segurança. Na verdade, a segurança deve fazer parte da natureza de um dispositivo IoT.

Nesse capítulo serão abordados alguns aspectos de segurança que devem ser utilizados em ambientes IoT e que foram implementados nesse projeto, visando garantir a confiabilidade do sistema.

4.1 Aspectos de Segurança

No ambiente IoT, geralmente as questões relacionadas com a segurança são negligenciadas, seja pela limitação de *hardware*, pelas dificuldades técnicas envolvidas ou por se julgar desnecessário. A grande quantidade de tecnologias que podem ser utilizadas também dificultam a determinação de uma infraestrutura padronizada [41], sendo necessário que se avalie cada caso.

Em 2018 foi publicado pelo *Open Web Application Security Project* (OWASP) [42] o *Top Ten* das vulnerabilidades mais comuns em IoT, dentre as quais podem ser destacadas as que envolvem a transferência de dados sem criptografia e dispositivos fisicamente vulneráveis. Com a preocupação de melhorar a segurança em dispositivos IoT houve um trabalho desenvolvido pelo *National Institute of Standards and Technology* (NIST) dos Estados Unidos, que buscou definir quais orientações as empresas devem seguir para coibir ataques a esses tipos de dispositivos [43].

Em 12 de maio de 2022 o presidente dos EUA, Joe Biden, assinou a Ordem Executiva 14028, resultante do projeto que foi desenvolvido pelo NIST, para apoiar a segurança cibernética, proteger a infraestrutura crítica e as redes do governo

federal [44].

De acordo com as orientações da norma IEEE 2413-2019 [1], os aspectos de segurança dos dispositivos devem garantir a integridade dos dados. Com isso, no trabalho desenvolvido nesta dissertação foram implementadas formas de mitigar ataques que envolvam a interceptação de dados e a alteração do *firmware* em dispositivos IoT que utilizam o ESP32. É importante observar que as implementações desenvolvidas apenas acrescentam novas camadas de segurança, mas existem ataques mais sofisticados que podem ser feitos em laboratórios, como os ataques de *Fault Injection and eFuse protection* [45], *Fault Injection and Secure Boot* [46], *Zero PMK Installation* [47], *EAP DoS* [48] e *Secure Boot Bypass* [49].

Em um ataque do tipo *Man-in-the-middle* (Figura 4.1), o invasor busca interceptar a comunicação do dispositivo com o servidor [50], para que as informações enviadas sejam adulteradas ou descobertas, de forma passiva (*Eavesdropping*), possibilitando que o atacante amplie o controle da rede através de técnicas de pivoteamento ou de movimento lateral. No ambiente de IoT essa é uma vulnerabilidade passível de ser explorada porque, em geral, os dados enviados pelos dispositivos são considerados como de baixa criticidade. Com isso, é comum que os usuários não se preocupem com a criptografia da informação transmitida devido ao fato de geralmente os dispositivos utilizados possuírem limitações de *hardware*. Essas limitações tornam mais complexo o desenvolvimento de soluções e dificultam a implementação de metodologias seguras, em razão de essas exigirem mais recursos de processamento e memória.

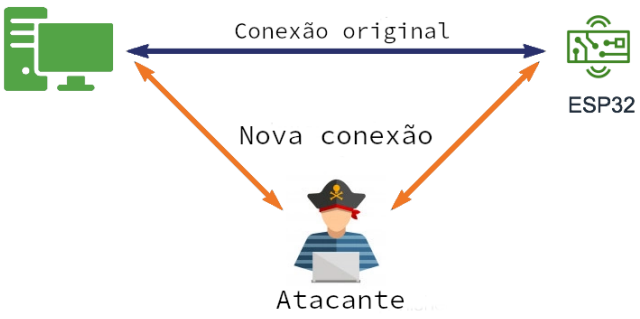


Figura 4.1: Ataque *Man-in-the-middle*.

Além da segurança associada ao dispositivo, também existe a necessidade de se manter seguro o servidor que faz a comunicação. A escolha do protocolo de comunicação é outra tarefa crítica no sistema, pois ela deve levar em consideração a

complexidade para implementação, taxa de transferência necessária e quantidade de dados enviadas por cada dispositivo. A escolha mais comum para os projetos IoT acaba sendo o protocolo *Message Queuing Telemetry Transport* (MQTT) ao invés do tradicional HTTP, pois é um protocolo que possui a natureza de ser utilizado para a coleta de dados entre muitos dispositivos que se conectam a uma infraestrutura computacional para processamento e armazenamento [51].

A característica que torna o MQTT um protocolo amplamente utilizado é o fato de ele necessitar de uma quantidade menor de pacotes para realizar e manter a comunicação. Porém, foi verificado [52] que para sistemas onde a conexão entre os dispositivos e o servidor não são reaproveitadas (os dados não são enviados na forma de *stream*), o custo no tráfego acaba sendo muito alto, tornando o protocolo menos eficiente que o HTTP. Sendo assim, devido a natureza das conexões entre os dispositivos de interesse serem de forma intermitente, a utilização de um protocolo com arquitetura *publish/subscribe* não é relevante para a grande maioria dos projetos desenvolvidos pelo laboratório. O protocolo HTTP foi o escolhido por também possuir suporte a comunicações seguras através de criptografia TLS (HTTPS), além de ser compatível com outros sistemas e aplicações.

A comunicação entre o ESP32 e o servidor é estabelecida através da conexão Wi-Fi, com autenticação baseada na especificação WPA2-PSK. O protocolo de criptografia escolhido foi o TLS v1.2, por ser amplamente suportado pelos sistemas e por existirem muitas bibliotecas disponíveis para o ESP32.

Para captura e análise dos dados de rede foram utilizadas os *softwares* tcpdump [53] e Wireshark [54].

4.2 Criptografia na comunicação

Para captura e análise dos dados de rede foram utilizadas os *softwares* tcpdump [53] e Wireshark [54]. As interfaces de desenvolvimento utilizadas (IDE) foram a do Arduino e o esp-idf.

Inicialmente, para comparação, a comunicação entre os dispositivos foi feita utilizando o protocolo HTTP e todo o tráfego da comunicação foi interceptada com o software *tcpdump*. A segurança da comunicação, nesse caso, fica dependente da rede e de suas políticas de acesso.

Na Figura 4.2 é possível observar o resultado da interceptação, com o fluxo da informação não criptografada, sendo possível identificar quando os dados são enviados (POST) e a rota utilizada (/medidores/).

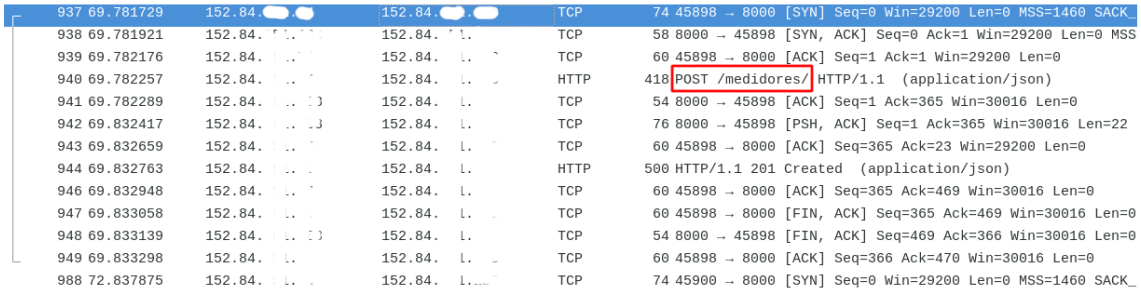


Figura 4.2: Interceptação da comunicação não criptografada (HTTP) entre o ESP32 e o servidor, utilizando o Wireshark.

Ao analisar o fluxo, fica claro a facilidade de recuperar informações sensíveis, como o formato em que os dados são enviados (JSON), o token de validação e as informações enviadas (Figura 4.3).

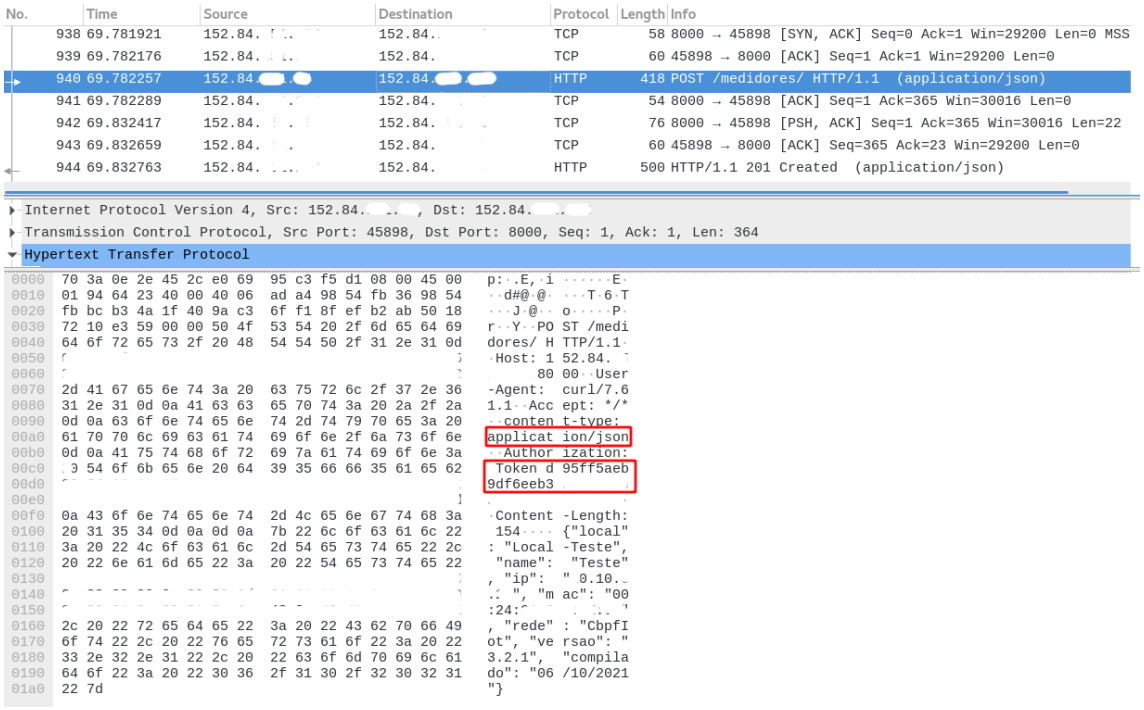


Figura 4.3: Dados não criptografados enviados pelo ESP32 para a API.

A programação para a comunicação através do protocolo HTTPS foi implementada de duas formas: a primeira foi com a utilização da IDE do ESP32 [55] (esp-idf), utilizando a biblioteca mbedtls [56]; a segunda foi através da biblioteca WiFiClientSecure [57], utilizando a interface do Arduino.

Com a comunicação segura foi possível observar que o tráfego de dados é feito de forma criptografada, com a utilização do protocolo TLSv1.2. No fluxo de dados que está na Figura 4.4 fica clara a diferença em relação aos dados não criptografados (Figura 4.3), sendo possível notar a etapa do *handshake* da comunicação entre o

ESP32 e o servidor.

Analisando o fluxo *Application Data* da Figura 4.5 também fica demostrada a melhoria na segurança, já que não é mais possível identificar informações sensíveis na comunicação em relação ao que está na Figura 4.3.

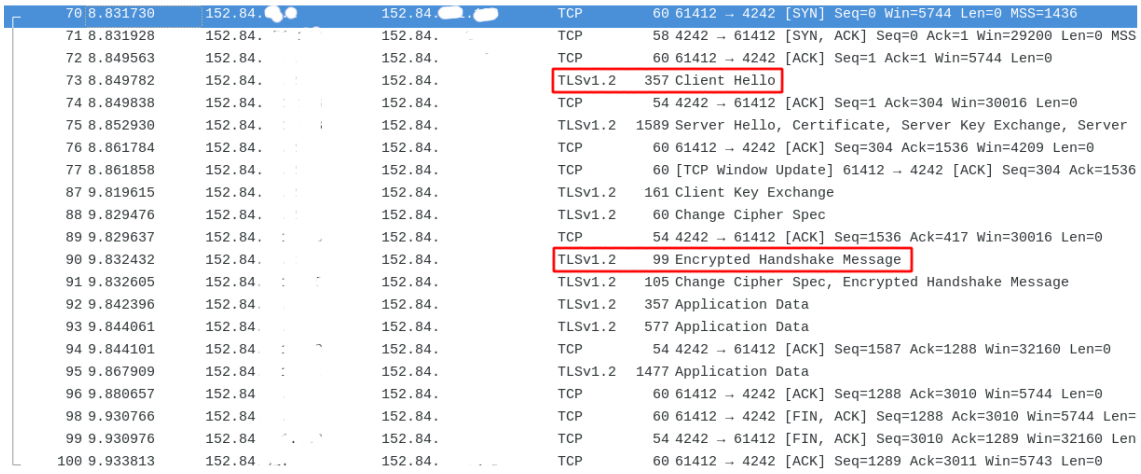


Figura 4.4: Interceptação da comunicação criptografada (HTTPS) entre o ESP32 e o servidor, utilizando o Wireshark.

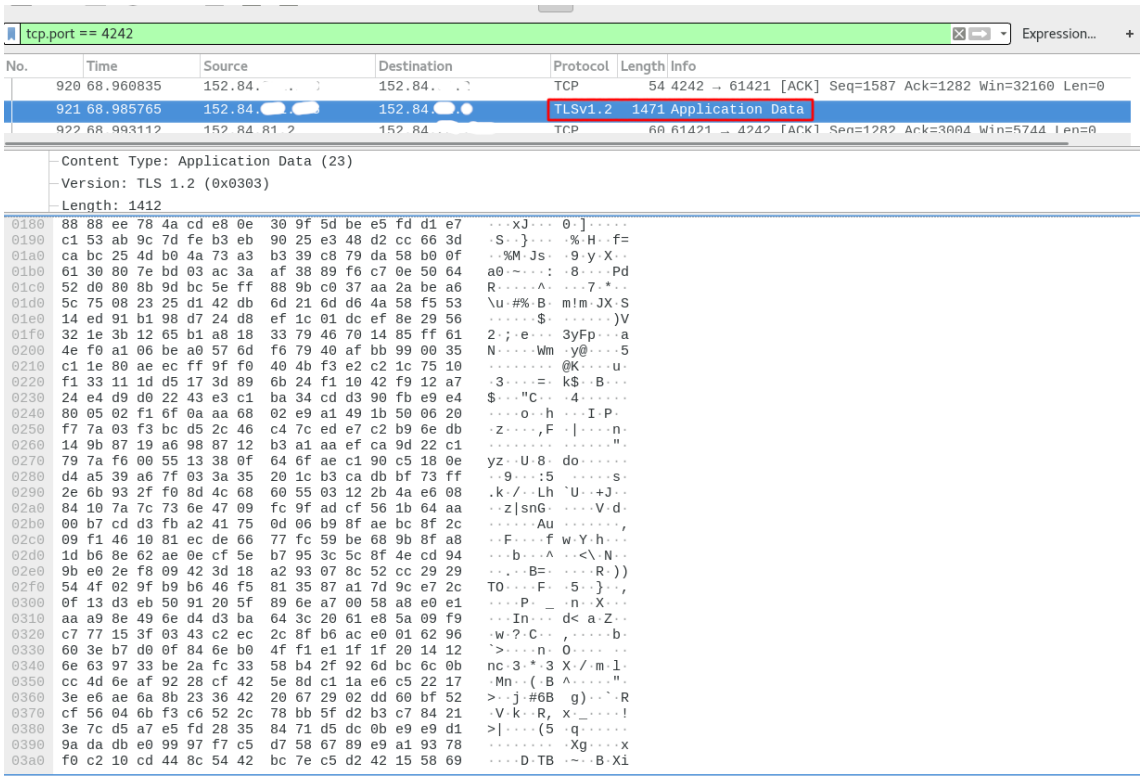


Figura 4.5: Dados criptografados através do protocolo TLSv1.2.

4.3 Criptografia do firmware

A taxonomia de uma vulnerabilidade de *hardware* pode ser divida entre a sua natureza e o seu domínio [58], como pode ser visto na Figura 4.6.

Caso uma vulnerabilidade de *hardware* possua natureza intencional, significa que ela foi implantada pelo fabricante/desenvolvedor de forma a permitir o acesso ao equipamento. Caso essa natureza não seja intencional, ela pode ser devido a falha do equipamento ou devido a lógica não segura de implementação (*bug*).

Por outro lado, uma vulnerabilidade de domínio lógico é implementada na fase inicial do projeto, enquanto a do domínio físico é implantada no final do projeto, na parte de design do equipamento.

Dentre as diversas categorias de ameaças, o objetivo com a criptografia do *firmware* é mitigar os ataques de Engenharia reversa e de modificação do *firmware* [59] que estão relacionadas a vulnerabilidades no domínio físico, onde um atacante que possua acesso ao *hardware* seja capaz de modificar o *firmware* e colocar um outro dispositivo semelhante no lugar.

Como muitas vezes os dispositivos ficam instalados em ambientes isolados e sem vigilância, um invasor de posse do ESP32 pode fazer o *dump* dos dados da memória *flash* e ter acesso a informações sensíveis, além de ser capaz de gerar e implantar um novo *firmware* que pode se comportar como o original, porém alterando as informações enviadas. Por exemplo, um atacante pode mascarar o envio de dados de temperatura, fazendo com que o dispositivo envie informações de que o ambiente está dentro dos limites de temperatura adequados, quando, na verdade, está ocorrendo um incêndio.

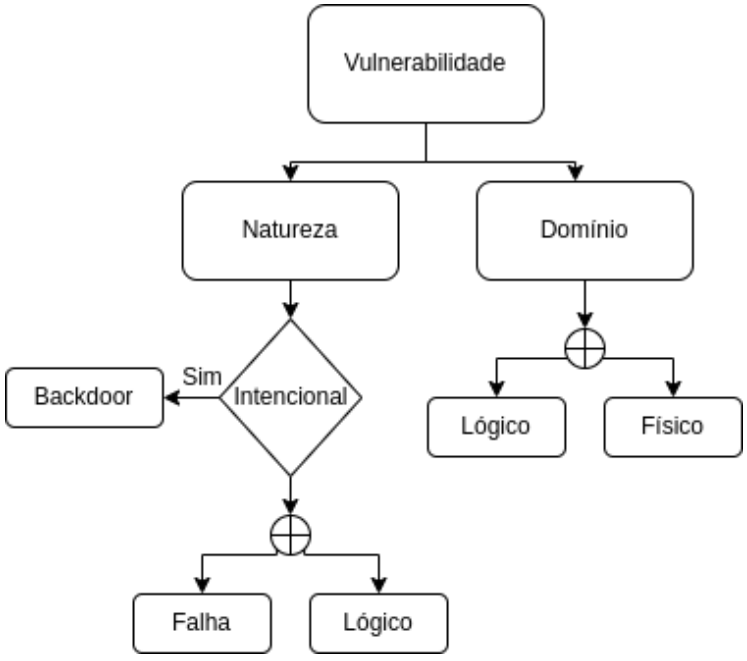


Figura 4.6: Taxonomia de uma vulnerabilidade de hardware.

Um exemplo de ataque foi o causado pelo *worm* Stuxnet [60]. A dinâmica envolvia infectar o sistema supervisório de uma usina de enriquecimento de urânio do Irã, que era executado em um sistema SCADA, mascarar os valores exibidos e acelerar as centrífugas utilizadas no projeto nuclear iraniano. Com isso foi possível danificar os equipamentos utilizados, prejudicando o programa nuclear daquele país.

4.4 Criptografia Flash e Inicialização Segura

A tecnologia utilizada no ESP32 para criptografia do *firmware* é baseada em fusíveis eletrônicos (eFuse), originalmente desenvolvida pela IBM [61], onde uma das aplicações é a de reprogramação dinâmica em tempo real de um chip.

O ESP32 possui um bloco de 1024 bits divididos em quatro eFuses com 256 bits cada - mas nem todos esses bits estão disponíveis - capazes de armazenar parâmetros do sistema e do usuário. Cada eFuse funciona como um bit que pode ser programado para 1, após o qual não pode ser reprogramado. No ESP32 é possível utilizar essa tecnologia para criptografar o *firmware* que será gravado no dispositivo [62].

Cada um desses blocos pode ser endereçado através das seguintes palavras reservadas:

- EFUSE_BLK0: Utilizado para propósitos gerais do sistema;
- EFUSE_BLK1: Utilizado para o armazenamento *flash* da chave criptografada. Pode ser utilizado para outros fins;
- EFUSE_BLK2: Utilizado para o *security boot key*. Pode ser utilizado para outros fins;
- EFUSE_BLK3: Pode ser parcialmente reservado para customização do endereço MAC ou utilizado completamente para alguma aplicação.

Na Figura 4.7 é possível ver a informação contida no firmware a partir do endereço 0001:0A40, não criptografada, obtida através da extração dos dados da memória. São visíveis os dados da chave utilizada para comunicação com a API, assim como o endereço do servidor e as rotas utilizadas.

Já na Figura 4.8 está o trecho com os dados para autenticação Wi-Fi. Por questões de segurança algumas partes das informações foram omitidas, mas é possível identificar o SSID no qual o ESP32 deve se conectar (CBPF-IoT), além de trechos da senha de acesso utilizada (-i2-7-).

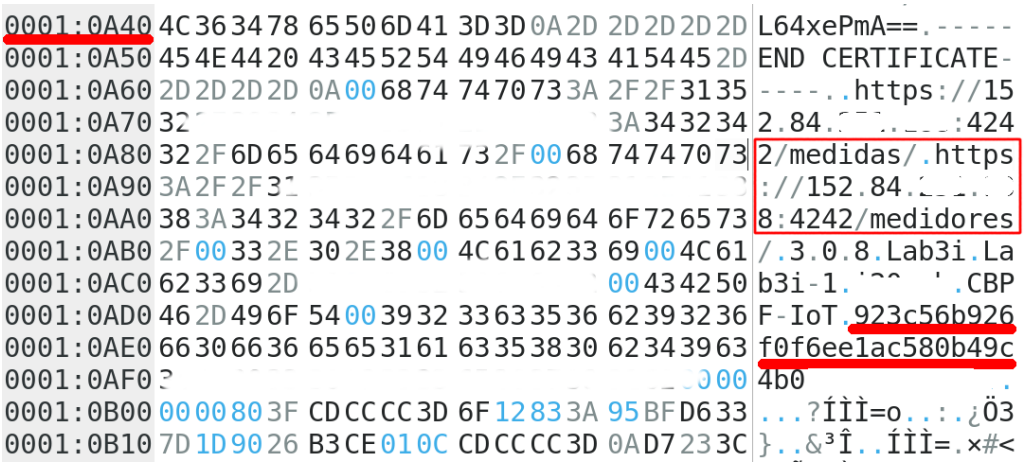


Figura 4.7: Trecho do *firmware* não criptografado com os dados da *API Key*.

Figura 4.8: Trecho do *firmware* não criptografado com os dados de acesso Wi-Fi.

O ESP32 possui mecanismos de segurança que permitem a criptografia do *firmware* através de seus eFuses [62].

O procedimento adotado nesse projeto foi o de gerar uma chave que deve ser gravada na memória do ESP32. Sendo assim, somente de posse dela será possível gerar um novo *firmware* capaz de ser executado no ESP32 (Figura 4.9).

Com isso é possível desenvolver o *firmware* utilizando a IDE do ESP32 ou a IDE do Arduino, sendo necessário apenas possuir o código binário gerado pelo compilador.

O procedimento de gravação da chave no *Efuse Block 1* faz com que o ESP32 fique bloqueado para a execução de *software* não assinado, com isso todo o *firmware* que for enviado deverá ser encriptado com a chave gerada anteriormente. Caso a chave seja perdida, não será mais possível executar um novo código no dispositivo.

Após a encriptação e gravação do código *firmware* no ESP32, foi executado o procedimento de extração e abertura do binário em um editor hexadecimal. O conteúdo pode ser visto nas Figuras 4.10 e 4.11. É possível verificar a efetividade do procedimento, pois o código foi gravado no *firmware* de maneira criptografada.

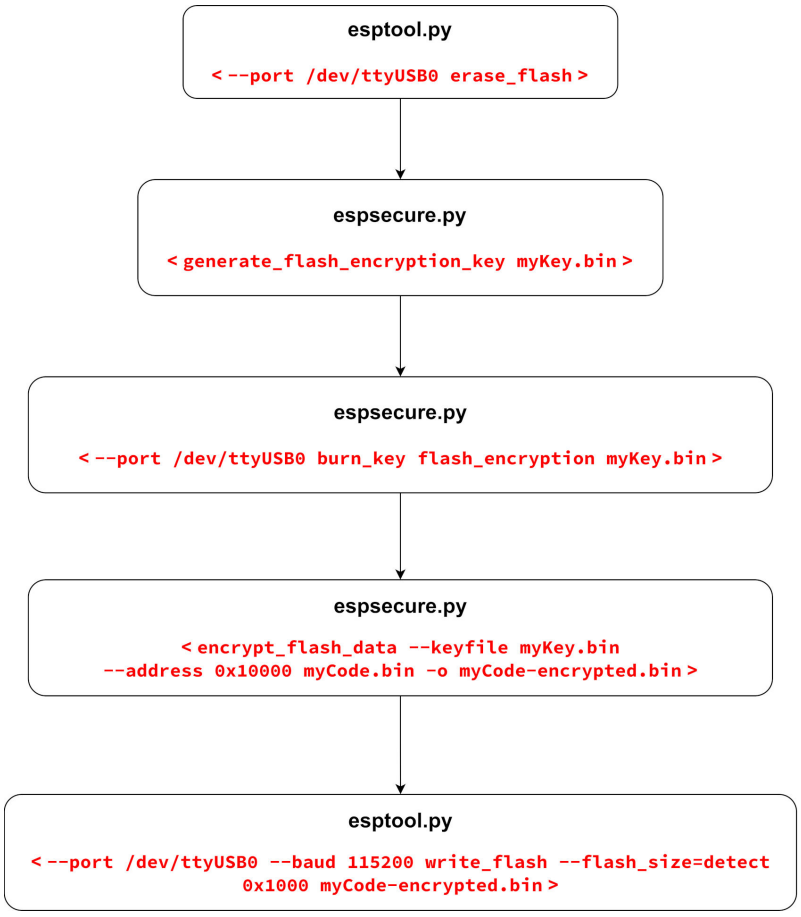


Figura 4.9: Fluxograma do procedimento de encriptação do *firmware*.

Nas Figuras 4.10 e 4.11 fica demostrada a efetividade do procedimento, pois o código foi salvo na memória *flash* de maneira criptografada.

As políticas desenvolvidas envolvendo a criptografia na comunicação e no *firmware* [63] fornecem um nível de segurança que deixam o projeto de acordo com as orientações estabelecidas no padrão IEEE 2413-2019.

Sendo assim, após finalizada esta etapa, o projeto possui os critérios de segurança e funcionalidade para ser instalado em ambientes reais de monitoramento, utilizando redes compartilhadas e com o dispositivo instalado em locais de acesso público.

No próximo capítulo serão discutidos os detalhes e os resultados do funcionamento da plataforma desenvolvida neste trabalho.

Figura 4.10: Trecho do *firmware* criptografado com os dados da *API Key*.

Figura 4.11: Trecho do *firmware* criptografado com os dados de acesso Wi-Fi.

Capítulo 5

Resultados e Discussões

Neste capítulo serão discutidos os aspectos mais relevantes que impactaram no desenvolvimento dessa dissertação. A organização é feita da seguinte forma: Testes de desempenho para identificação dos limites de operação da plataforma; Desenvolvimento do supervisor para monitoramento das variáveis coletadas; Monitoramento elétrico dos motores e ambiental da Criogenia do CBPF; Monitoramento do painel elétrico e do ambiente do Lab3I; Detecção da onda de choque do vulcão de Tonga, através da variação da pressão atmosférica; Monitoramento do ambiente de obras raras, contendo livros e periódico com mais de 150 anos; Monitoramento do laboratório contêiner do Projeto Neutrinos Angra; Desafios de implementação.

5.1 Testes de desempenho

Dois critérios do projeto envolvem a capacidade de escalabilidade e de gerenciamento de grandes volumes de dados (Capítulo 2). Com o intuito de verificar o comportamento do sistema e avaliar os limites de operação foi utilizada a ferramenta Apache JMeter [64]. O JMeter é capaz de realizar testes de carga e de estresse em sistemas computacionais.

A ferramenta foi configurada para que fosse feita uma simulação de requisições ao servidor, com o comportamento equivalente ao de vários equipamentos, efetuando a conexão e o envio de dados ao servidor. Foi definido que a quantidade de requisições atingiria o limite quando o servidor passasse a rejeitar as requisições.

Conforme detalhado no Apêndice B, o método POST do equipamento faz o envio de 6 atributos a cada inicialização e 26 atributos a cada 5 segundos. O *payload* é de aproximadamente 180 bytes e 500 bytes respectivamente.

Na Figura 5.1 está o resultado referente aos valores médios de requisições por minuto. Na configuração utilizada o sistema foi capaz de responder a aproximadamente

95 requisições por segundo, ou seja, o equivalente a 95 dispositivos se conectando ao servidor a cada segundo.

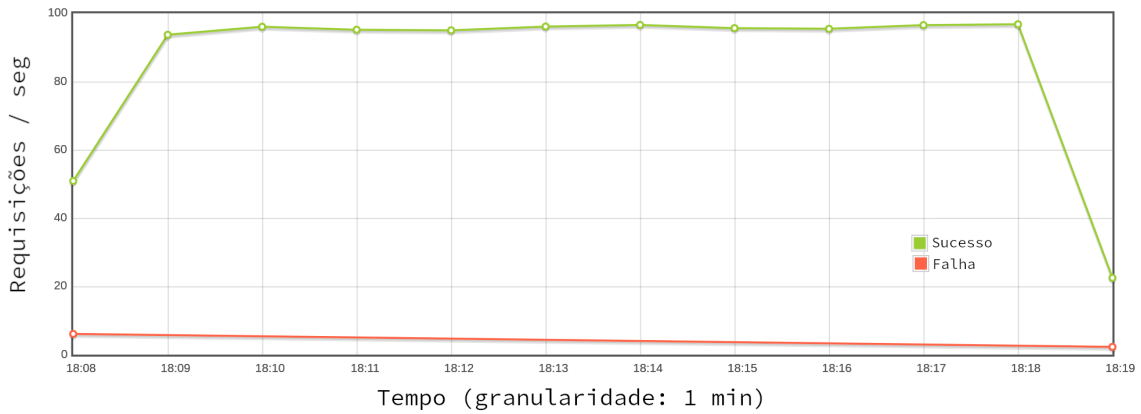


Figura 5.1: Quantidade média de requisições por segundo para até 100 dispositivos, com granularidade em minutos.

Esse foi o limite para o qual o sistema começou a rejeitar as conexões. Dentro do JMeter foi configurado para que o número de requisições fosse diminuindo assim que o servidor passasse a rejeitar as conexões, até chegar a um limite onde as requisições fossem atendidas (Figura 5.1).

As rampas no início e no fim do teste são devido ao aumento e diminuição no número de requisições do teste no primeiro e último minuto, respectivamente.

Também foi avaliado o tempo de resposta para checar qual seria o intervalo limite entre as requisições. O tempo médio ficou abaixo de 1500 ms, conforme pode ser visto do histograma da Figura 5.2.

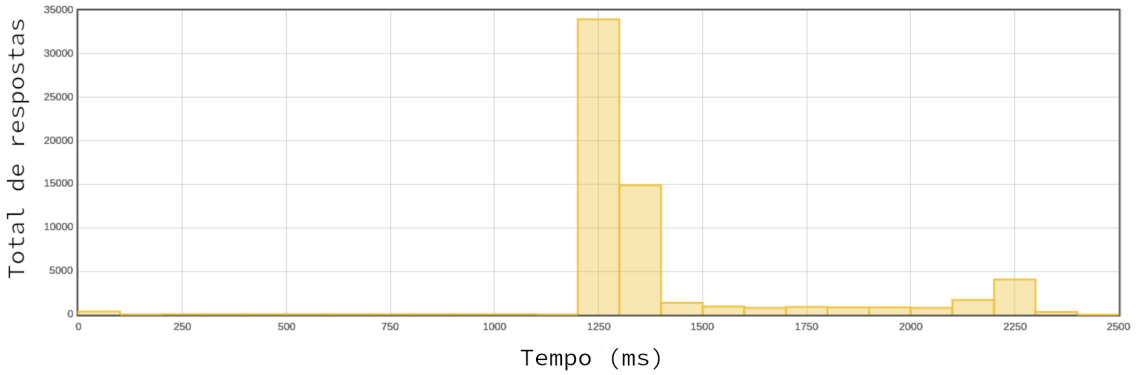


Figura 5.2: Frequência relativa dos tempos médios de resposta.

Com os dados da Figura 5.1, a conclusão é que para a estrutura atual do sistema se comportar de maneira estável é necessário que o número máximo de requisições por segundo ao servidor fique abaixo de 90 req/seg. Já na Figura 5.2 é possível ver que nesse limite o tempo de resposta do servidor irá ficar em aproximadamente 1250

ms, o que pode se tornar um problema caso o intervalo entre as medições seja muito pequeno.

5.2 Supervisor

O supervisor é um ambiente que permite monitorar o que acontece com o sistema de forma remota.

O ambiente é composto essencialmente por 3 áreas (Figura 5.3). Na área 1 fica a identificação do sistema, do usuário que está logado, os menus aos quais o usuário possui acesso, um ícone para acesso às configurações e um botão para sair do sistema.

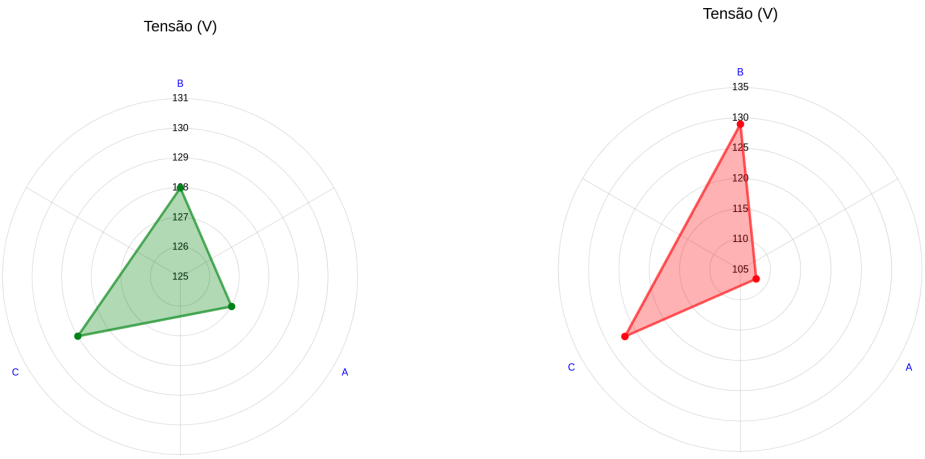
Na área 2 são exibidas algumas informações e alertas que o sistema pode enviar com base nos locais monitorados.

Na área 3 são exibidos os gráficos dos dados que são coletados.

Figura 5.3: Página inicial do sistema de supervisão.

Uma das características de um ambiente supervisor é permitir que o usuário seja capaz de perceber de forma rápida e visual a ocorrência de algum problema. Para a visualização das tensões uma das opções disponíveis envolve a interpretação geométrica. Dado que as tensões devem ser iguais por fase, a figura para um sistema em condições ideais de operação deve se aproximar a forma de um triângulo equilátero (Figura 5.4a).

Para diferenças de tensão, como no caso da Figura 5.4b, a representação do triângulo irá se deformar. Caso essa diferença fique fora dos parâmetros estabelecidos, a área do gráfico será preenchida pela cor vermelha.



(a) Fases equilibradas. (b) Fases desbalanceadas.

Figura 5.4: Gráfico do sistema para o monitoramento de fases através da representação geométrica.

O desenvolvimento do supervisor é útil para implementação de aplicações distintas, incluindo a possibilidade de acionamento de equipamentos e análise de dados.

5.3 Monitoramento elétrico na Criogenia

No início dos testes o protótipo foi instalado no setor de Criogenia do CBPF, para monitorar a operação de um Chiller. A Criogenia é o setor que possui as características de ambiente, equipamentos e consumo de energia mais próximos de um ambiente industrial. Esse setor é responsável pela produção de água gelada, Nitrogênio líquido, possui equipamentos para liquefação de Hélio, além de ser o local com a maior concentração de equipamentos com alto consumo de energia do CBPF. Nesse teste foram verificados os modos de operação autônoma (Figura 5.5) e de monitoramento (Figura 5.6).

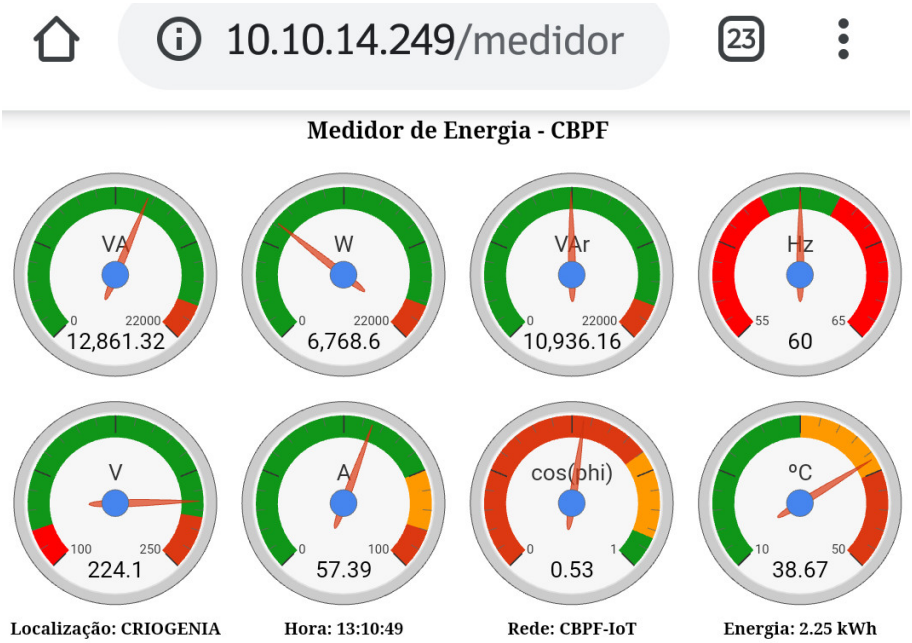


Figura 5.5: Modo de operação autônoma do equipamento em teste na Criogenia.

No modo de operação autônomo, onde o equipamento executa o seu próprio servidor web podendo ser acessado através do endereço IP, é possível ver as características de operação através de uma página web que é gerada e atualizada diretamente pelo equipamento (Módulo de Medição). O acesso pode ser feito através do endereço IP do equipamento com a rota "/medidor", conforme pode ser visto na Figura 5.5.

Já no modo de monitoramento, os dados foram enviados através da API e armazenados no banco de dados. Isso permite, no longo prazo, uma análise sobre a operação do equipamento. Na Figura (Figura 5.6) é possível observar o ciclo de atividades através do monitoramento da corrente elétrica.

As variações em degrau no gráfico são devido ao acionamento e desligamento do Chiller. Quando a temperatura da água atinge um valor máximo estabelecido, o sistema é ligado e baixa a temperatura da água até um valor mínimo estabelecido e então ele é desligado. Esse ciclo pode variar de acordo com a necessidade de utilização e variação térmica no dia.

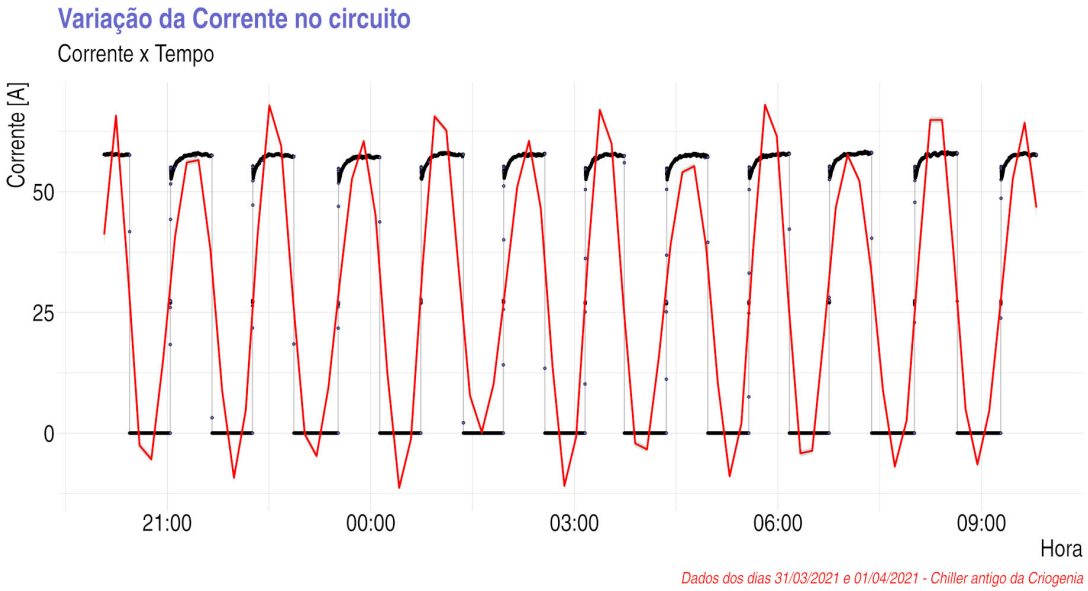


Figura 5.6: Variação da corrente elétrica durante a operação do Chiller.

5.4 Monitoramento elétrico e ambiental no Lab3I

O Laboratório de Informação, Instrumentação e IoT (Lab3I) está situado na biblioteca e faz parte do Núcleo de Informação e Biblioteca (NIB). O Lab3I foi planejado com a proposta de disseminar, através do ensino e pesquisa, a cultura necessária para que as pessoas sejam capazes de entender e utilizar a tecnologia da Internet das Coisas.

Após a etapa de validação, o sistema foi instalado no Lab3I - como pode ser visto na Figura 5.7. No laboratório o equipamento ficou em testes por aproximadamente 6 meses. Nesse período foram feitas melhorias para estabilidade do sistema e otimizações das rotinas no código.

As variáveis ambientais escolhidas para monitoramento foram temperatura, pressão atmosférica e umidade, pois essas grandezas geralmente impactam no modo de operação dos equipamentos e nos resultados de experimentos.

A seguir serão apresentados os dados coletados pelos sensores durante o período que compreende os dias 21 de outubro de 2021 a 2 de fevereiro de 2022 . Para esta configuração o equipamento fez a medição e o envio das três grandezas a cada 5 segundos.

Figura 5.7: Equipamento instalado em um quadro de alimentação.

Nas figuras a seguir são representados os gráficos dos pontos com os valores medidos para a temperatura (Figura 5.8), umidade (Figura 5.9) e pressão atmosférica (Figura 5.10). As linhas em vermelho são curvas de tendência para guiar a visualização.

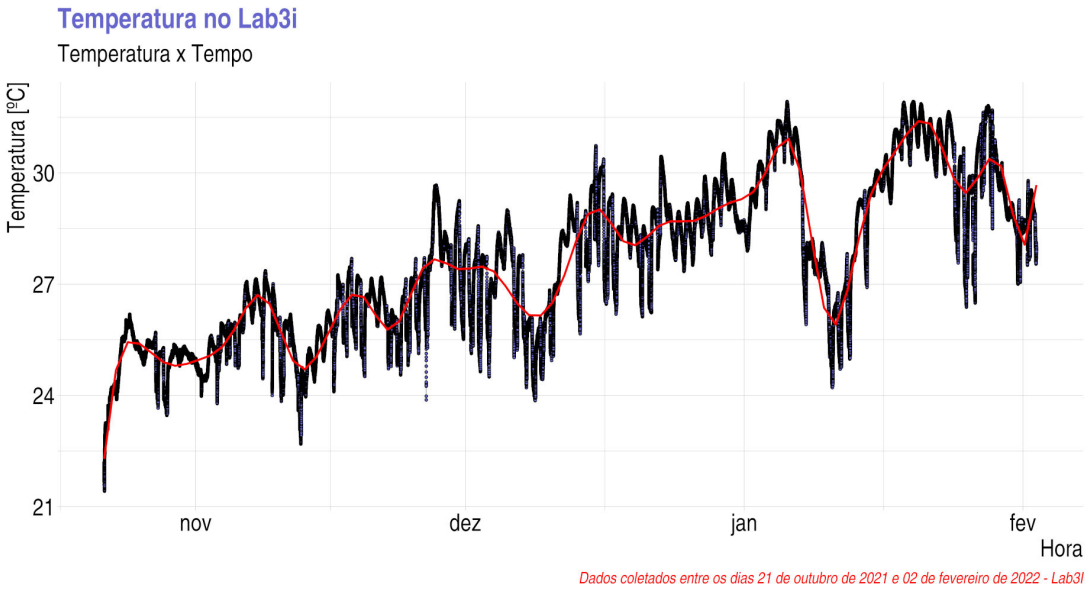


Figura 5.8: Variação da temperatura ao longo do tempo no Lab3I, CBPF.

Com base no gráfico da Figura 5.8 é possível avaliar como foi a variação de temperatura no laboratório entre os meses de outubro de 2021 e fevereiro de 2022.

Segundo a NBR-16401-2:2008 [65] a temperatura deve permanecer na faixa entre $22,5^oC$ e $25,5^oC$ para as condições de trabalho em um laboratório. Para o ambiente em questão, essa variação um pouco fora da margem estabelecida pode ter sido causada pela localização do sensor de temperatura. Como essas medições foram feitas em um período de testes, o sensor foi posicionado em cima do quadro de alimentação elétrica que, por sua vez, fica próximo a janela. O fato do ar condicionado também se manter em operação somente quando o laboratório está em utilização pode ter influenciado nessas medições.

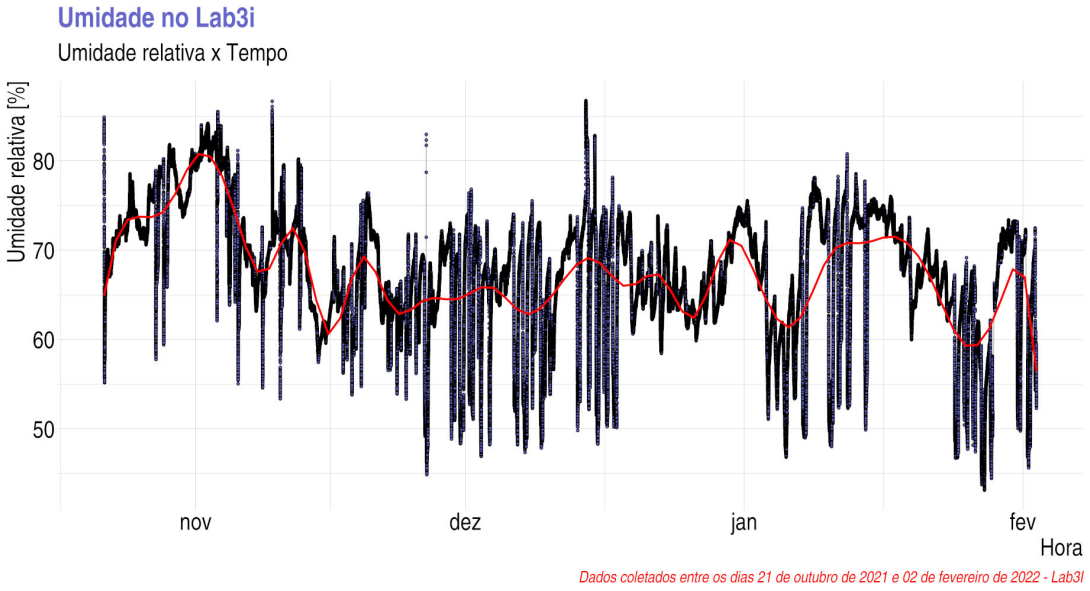


Figura 5.9: Variação da umidade ao longo do tempo no Lab3I, CBPF.

Ainda segundo a Norma [65], as condições de umidade relativa do ar também devem permanecer em 65%, para operação no verão.

Os dados das medições coletadas pelo equipamento são armazenados em um banco de dados na nuvem para posterior recuperação e análise.

Outra característica é a possibilidade de integração com outras ferramenta de visualização interativa, como o Grafana [66] ou o Zabbix [67]. Na Figura 5.11 está um *dashboard* elaborado com os dados coletados do painel elétrico do Lab3I.

Através dessa outra implementação é possível combinar a visualização de várias grandezas em uma única tela, de forma interativa e em tempo real. Ainda na Figura 5.11 é possível observar os valores medidos de umidade, temperatura, pressão atmosférica, tensão, corrente, potência, fator de potência ativa e frequência da rede em cada uma das fases de alimentação.

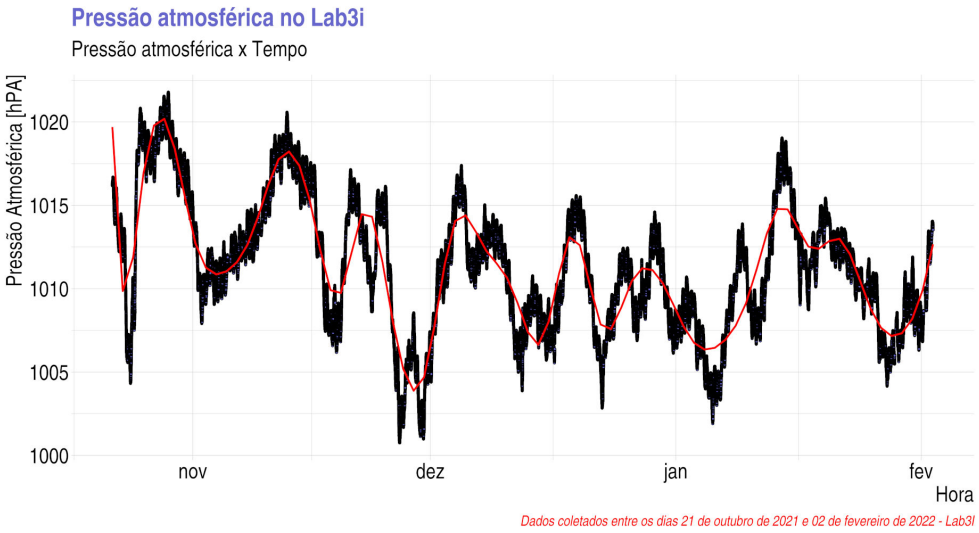


Figura 5.10: Variação da pressão atmosférica ao longo do tempo no Lab3I, CBPF.

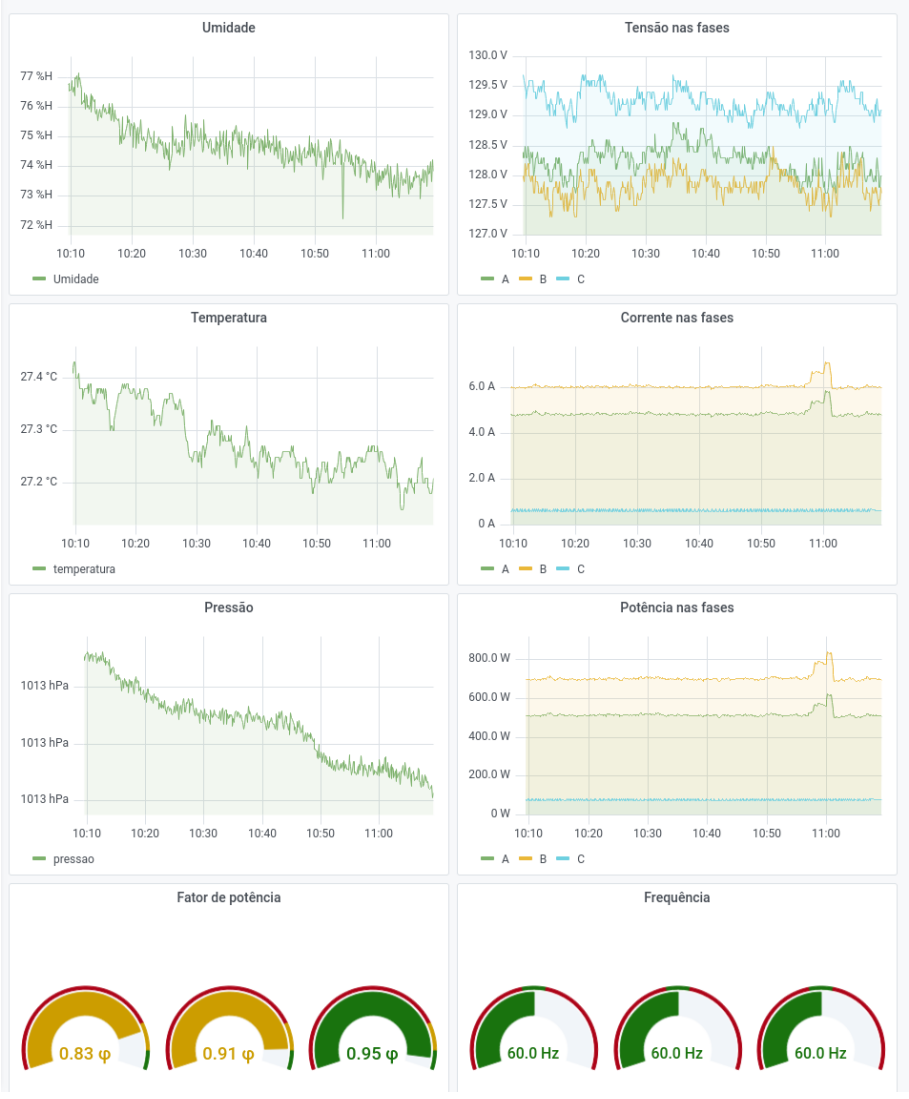


Figura 5.11: Dashboard criado no Grafana para monitoramento do painel elétrico.

5.5 Detecção da onda de choque do vulcão de Tonga

Durante o período de aquisição dos dados ocorreu a erupção do vulcão Hunga Tonga-Hunga Ha'apai, na nação polinésia de Tonga no Pacífico Sul [68]. A onda de choque dessa erupção foi detectada em várias partes do mundo, inclusive no Brasil [69], e amplamente divulgado na mídia na época.

Ao analisar os dados coletados pelo sensor para o período compatível com as detecções feitas no Brasil (entre as 12h e 13h do dia 15 de janeiro de 2022, horário de Brasília), foi possível verificar que os instrumentos utilizados foram capazes de medir a onda de choque como variação na pressão atmosférica [70], como pode ser visto na Figura 5.12.

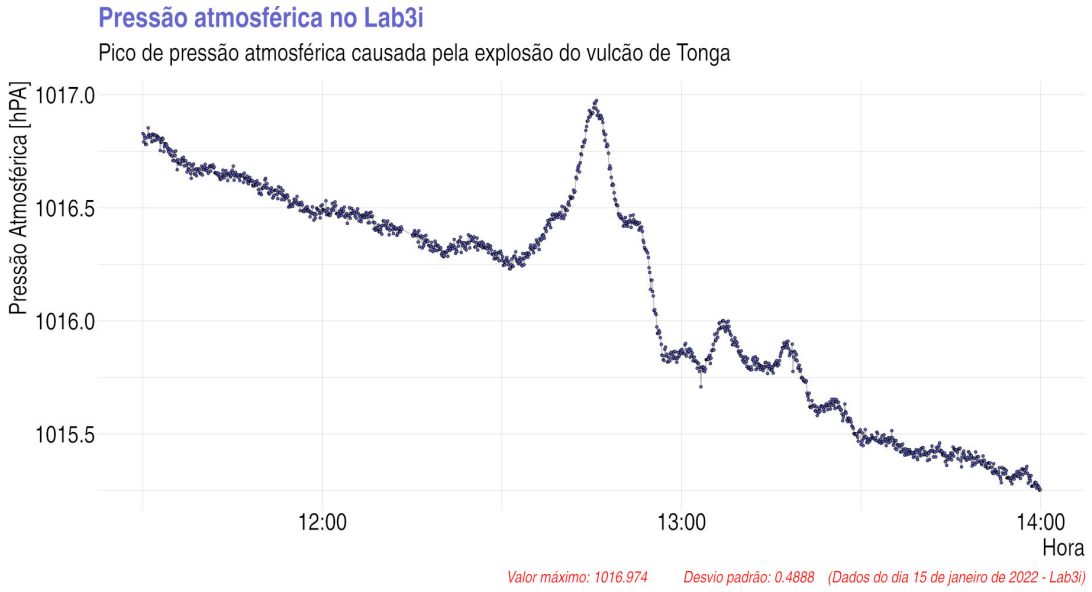


Figura 5.12: Variação na pressão atmosférica devido a erupção do vulcão em Tonga, medido no Lab3I, CBPF.

Para a análise desse evento foi criado um script na linguagem R que permite a análise de dados de um sensor específico, conectado a um sistema de medição identificado pelo seu endereço MAC, em uma janela de tempo. Isso otimizou o processo de acesso ao banco de dados e poderá ser utilizado quando houver a necessidade de novas análises específicas para um evento.

5.6 Aplicação no Monitoramento do ambiente de obras raras

O CBPF possui um acervo com cerca de 31.900 livros e mais de 900 títulos somente de periódicos, contendo publicações desde o século XVIII. Nessa coleção,

existem sete títulos que o Catálogo Coletivo Nacional (CCN), veiculado pelo Instituto Brasileiro de Informação em Ciência e Tecnologia (IBICT), considera como patrimônios nacionais unicamente disponíveis no CBPF.

Dentre as coleções se destacam as de periódicos de grande importância histórica, como *The Philosophical Magazine* (1800), *Comptes Rendus des Séances de L'Académie des Sciences* (1835), *Philosophical Transactions of the Royal Society of London* (1875), e *Proceedings of the Royal Society of London* (1877). Com isso, a conservação desse acervo é fundamental para a manutenção desse patrimônio nacional.

O monitoramento das grandezas umidade relativa do ar e temperatura são essenciais, já que essas variáveis contribuem diretamente para a deterioração do acervo, além de favorecerem a proliferação de agentes biológicos nocivos.

Idealmente, os valores de temperatura e umidade devem permanecer entre 19^{o}C e 23^{o}C e 50% e 60%, respectivamente [71].

Atualmente o ambiente é monitorado pelo sistema desenvolvido (Figura 5.13) e auxilia o corpo técnico na visualização do histórico de temperatura e umidade, além de facilitar a detecção de qualquer anomalia que pode estar relacionada com o funcionamento do sistema de climatização.

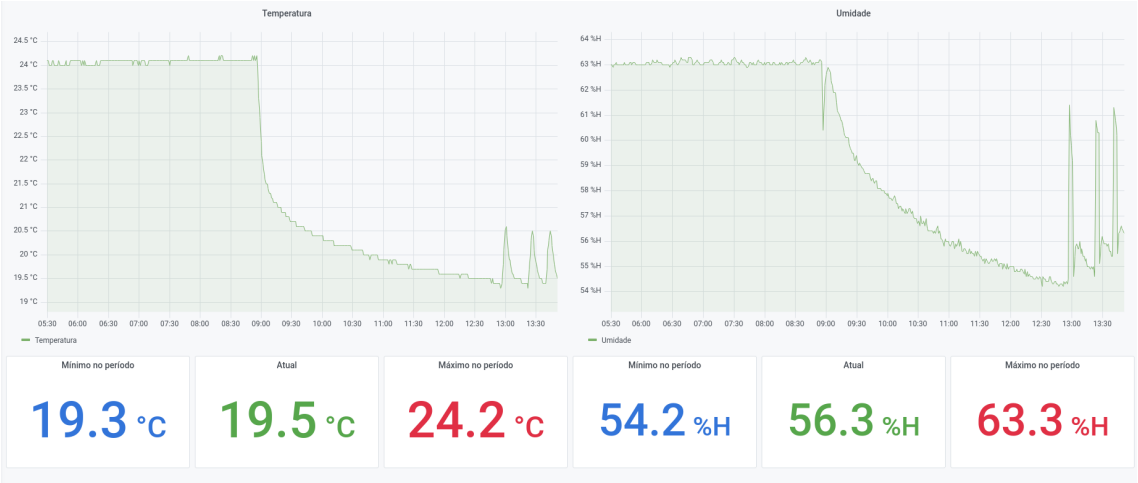


Figura 5.13: Painel de monitoramento do acervo utilizando.

Na janela com o gráfico (Figura 5.13) é possível selecionar o período e na parte de baixo são mostrados os valores mínimos (azul) e máximos (vermelho) do período especificado, além dos valores medidos atualmente (verde).

5.7 Aplicação no Projeto Neutrinos Angra

O Experimento Neutrinos Angra foi desenvolvido com o propósito de criar um detector de antineutrinos com capacidade de monitorar parâmetros relacionados à

atividade de reatores nucleares, como a composição isotópica do combustível nuclear e a potência térmica instantânea liberada pelo reator. Essa aplicação de física de neutrinos está de acordo com os propósitos do programa da Agência Internacional de Energia Atômica (AIEA) de desenvolvimento de novas técnicas para controle e aplicação de salvaguardas para não-proliferação de armas nucleares [72] .

O detector fica localizado em um contêiner adaptado para laboratório, ao lado da cúpula de blindagem do reator de Angra II (Figura 5.14), a 25 metros do núcleo do reator, podendo detectar até 5400 interações de antineutrinos/m^3 de água por dia, de acordo com simulações [73] [74].

Como os pesquisadores envolvidos no projeto trabalham e residem em cidades distantes da usina, a operação dos experimentos é feita de forma totalmente remota. Com isso, as variáveis monitoradas precisam ser enviadas periodicamente via rede de dados para o CBPF a fim do acompanhamento contínuo da situação no contêiner, garantido a segurança do laboratório e o bom andamento dos experimentos. A utilização do sistema desenvolvido nessa dissertação irá ajudar no monitoramento do que acontece dentro do contêiner para identificação de algum problema, como o aumento da temperatura devido a falha no sistema de climatização, entrada de luz e trânsito de pessoas, por exemplo.

Figura 5.14: Localização do contêiner Neutrinos. (Crédito: Herman P. Lima)

A técnica utilizada para detecção é baseada na radiação Cherenkov . Essa radiação é emitida por uma partícula carregada que atravessa um determinado meio com uma velocidade superior à velocidade da luz nesse meio [75]. A radiação eletromagnética emitida pode ser vista na faixa do espectro visível como uma luz de aspecto azulada.

Atualmente, a Colaboração Neutrinos Angra é composta por pesquisadores e

estudantes de 6 instituições brasileiras: Centro Brasileiro de Pesquisas Físicas - Rio de Janeiro-RJ, Universidade Estadual de Feira de Santana - Feira de Santana-BA, Universidade Estadual de Londrina, Londrina-PR, Universidade Federal da Bahia, Salvador-BA, Universidade Federal de Juiz de Fora, Juiz de Fora-MG e Universidade Estadual de Campinas, Campinas-SP.

O projeto para o monitoramento do contêiner foi adaptado para que os sensores operem de forma segura e independente do sistema já existente. Nesse caso foi utilizado o Raspberry Pi como centro de coleta e processamento de borda, ao invés de um servidor em nuvem. A conexão com a rede da usina possui muitas restrições de acesso, visando garantir a segurança e a disponibilidade do tráfego. O Raspberry foi configurado com a distribuição Raspberry Pi OS e a comunicação com o servidor é feita através da rede cabeada, com os dispositivos ESP32 se comunicando através da rede Wi-Fi diretamente com o Raspberry Pi.

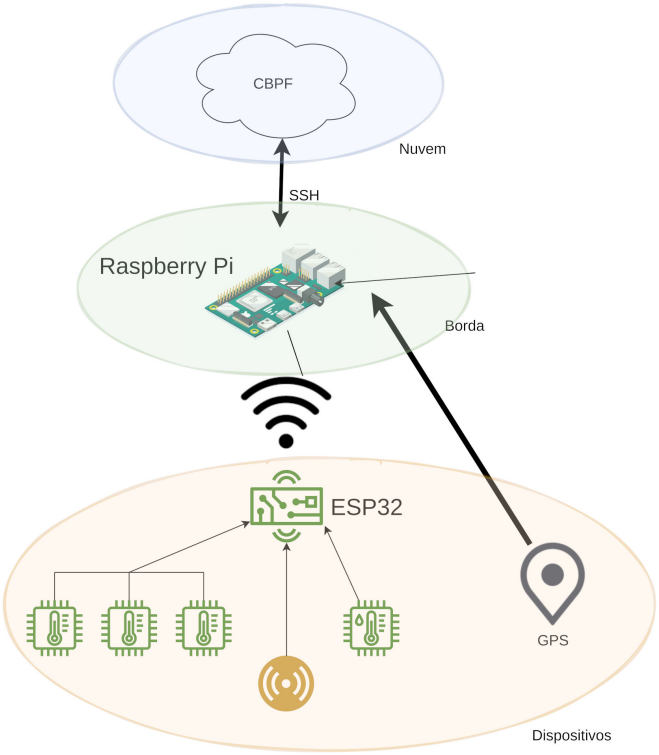


Figura 5.15: Topologia para monitoramento do contêiner.

Com isso, o projeto foi adaptado para uma arquitetura de três camadas, composta por uma Camada de Nuvem, uma Camada de Borda e uma Camada de Dispositivos [28]. Na Camada de Nuvem estão os servidores locais e a Nuvem do CBPF, na Camada de Borda está o Raspberry Pi e na Camada de Dispositivos está o equipamento com os sensores conectados ao ESP32.

O equipamento foi configurado para ser utilizado com sensores de temperatura; temperatura e umidade; temperatura, umidade e pressão atmosférica; luminosidade;

presença; medição de energia. Todos esses sensores já fazem parte da biblioteca desenvolvida (Apêndice A).

O sistema foi instalado no contêiner (Figura 5.16) para monitorar a presença de pessoas, umidade, pressão atmosférica, temperatura em diversos pontos e luminosidade.

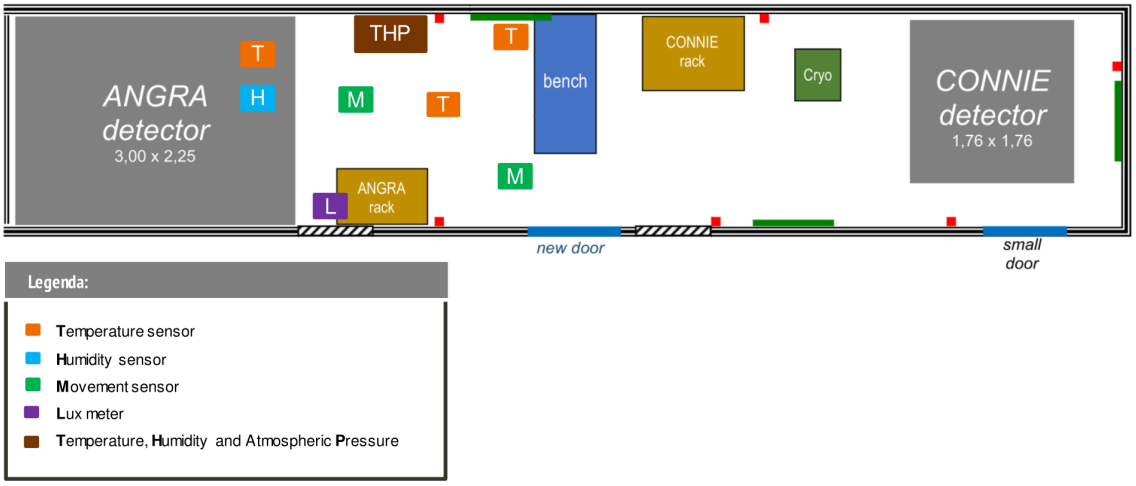


Figura 5.16: Representação do local de instalação.

Os dados serão armazenados localmente no Raspberry Pi e, posteriormente, serão sincronizados com servidores remotos. Localmente é possível monitorar o contêiner através da interface do Grafana.

Devido ao experimento estar localizado dentro do complexo da Usina Nuclear existem muitas restrições de acesso a rede. O canal de comunicação é restrito ao CBPF, não sendo possível acessar diretamente a Internet através da rede do contêiner.

O Raspberry Pi é um *hardware* versátil mas desenvolvido com a filosofia de *ultra-low cost computer* [76], devido a isso ele não possui um módulo *Real Time Clock* (RTC) alimentado por uma bateria. A sincronização da hora é feita através do sistema operacional a partir de servidores *Nework Time Protocol* (NTP) disponíveis na Internet. Esta condição compromete as medições feitas pelo equipamento pois seria impossível saber qual a hora certa em que os dados foram medidos pelos sensores, sendo necessária a sincronização manual da hora toda vez que o servidor fosse reiniciado.

Esse problema foi resolvido com a utilização de um módulo de recepção de sinal *Global Positioning System* (GPS) ligado ao Raspberry Pi. Esse módulo possui o chip NEO-M8N da empresa *u-blox* [77] e a comunicação é feita via porta USB (Figura 5.17).

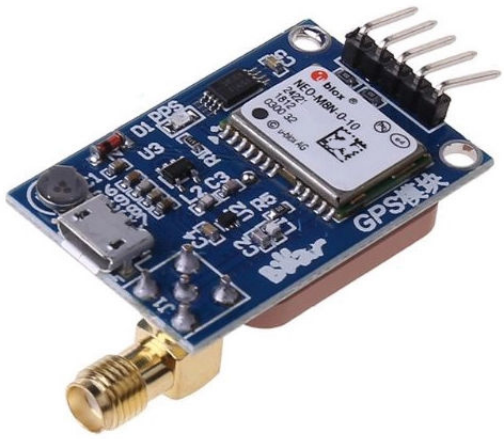


Figura 5.17: Módulo GPS NEO-M8N para sincronização da hora.

O GPS é um sistema de navegação baseado em satélites, desenvolvido pelo Departamento de Defesa dos Estados Unidos (DoD) no início dos anos de 1970. O sistema de GPS oferece, de forma contínua, informações sobre o posicionamento e a localização em qualquer lugar do mundo, sob quaisquer condições climáticas [78].

Após a conexão no Raspberry Pi o módulo foi reconhecido pelo Rocky Linux como um dispositivo serial na porta /dev/ttyACM0. Com isso foi possível configurar o sistema para utilizar o GPS para sincronizar a hora com uma precisão da ordem de Microsegundos.

Para aumentar a precisão foi ajustado o *offset* para as medições. Para isso o valor desse desvio, que já é fornecido pelo sistema de GPS, foi coletado durante aproximadamente um dia, a cada duas vezes por segundo (Figura 5.18).

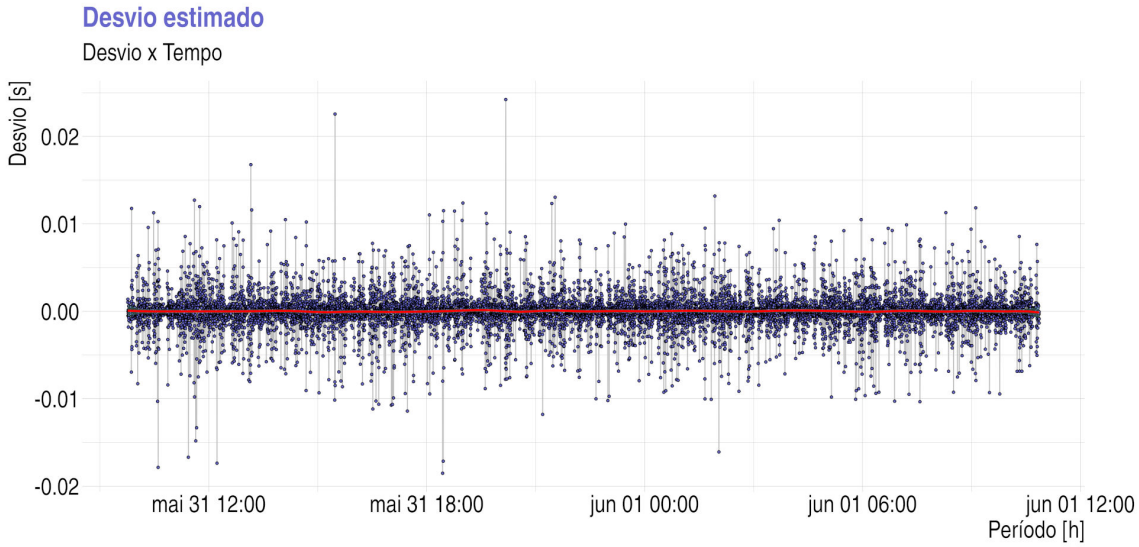


Figura 5.18: Desvio na hora coletada pelo GPS durante um dia.

O valor médio dos desvios ficou em $\approx 3,7$ ms, com o desvio padrão em

$\approx 0,002$ ms. Após o ajuste, o erro estimado para a medição da hora através do GPS ficou em $\approx 1,8$ ms. Existem diversos fatores que podem influenciar nesse erro, como dias chuvosos ou baixa visibilidade dos satélites. Durante o período em que essas medições foram feitas o tempo estava estável. Para a medição da posição é necessário a sincronização com ao menos quatro satélites: três para definição da posição e um adicional para o ajuste do erro na hora. Já para aquisição da hora, basta a sincronização com ao menos um satélite.

Essa implementação também foi útil para tornar o Raspberry Pi em um servidor de hora local, possibilitando a sincronização da hora por outros sistemas localizados na mesma rede, através do protocolo NTP.

Na Figura 5.19 é possível ver o *dashboard* em funcionamento no laboratório, pronto para ser instalado no ambiente do contêiner.

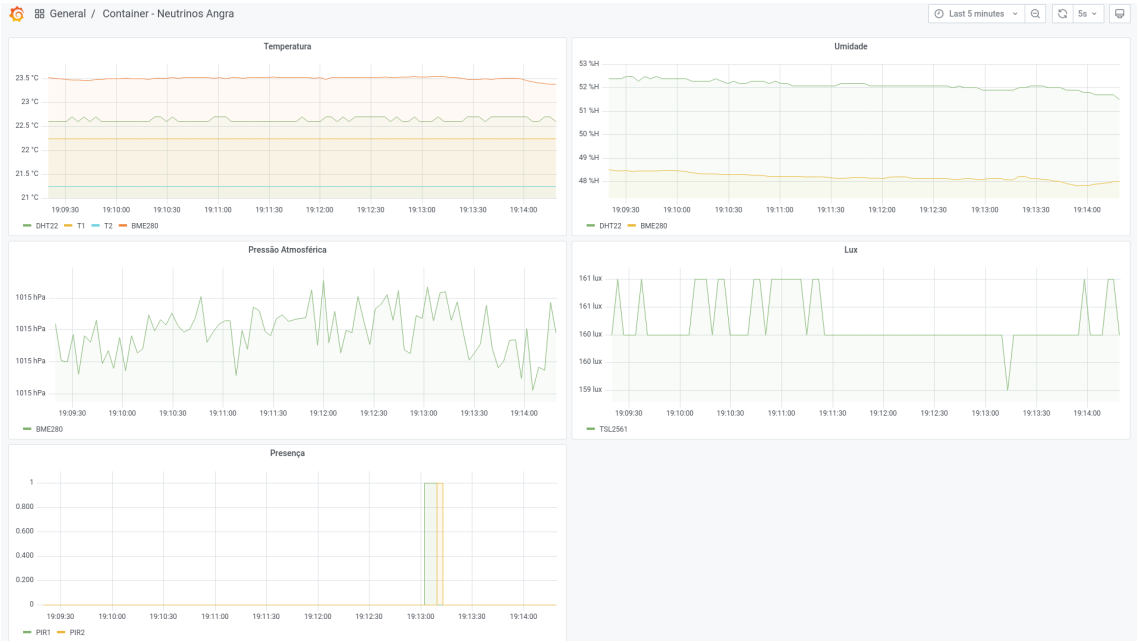


Figura 5.19: Visualização das variáveis monitoradas em teste no laboratório.

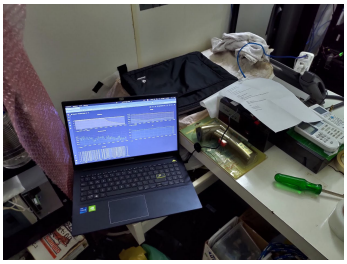
Ao final desse projeto e após uma janela de disponibilidade para acesso à Usina Nuclear, a instalação do equipamento foi feita no final do mês de agosto de 2022 (Figuras 5.20 e 5.21).

Como a única forma de transferir os dados coletados no contêiner para o CBPF é através de conexão por SSH, conforme pode ser visto no esquema da Figura 5.15, foi necessário a elaboração de uma ferramenta para automatizar esse processo.

(a) Instalação dos sensor de temperatura e umidade

(b) Instalação do sensor de luminosidade

Figura 5.20: Instalação dos sensores dentro do laboratório contêiner.



(a) Visualização dos dados capturados

(b) Localização do equipamento

Figura 5.21: Plataforma em funcionamento.

A cada intervalo de tempo determinado o *script* desenvolvido checa qual é a data do último dado armazenado no banco de dados PostgreSQL localizado no CBPF, acessa o Raspberry Pi localizado no contêiner e faz o *download* dos dados ainda não sincronizados. Após isso, os dados são gravados no banco de dados no CBPF para posterior visualização.

Além disso, também foi criado o site https://argus.cbpf.br baseado no Grafana [66], para monitoramento dos diversos locais onde o sistema esteja instalado. Atualmente o acesso ao sistema está disponível apenas através da Intranet do CBPF. O ambiente possui usuários independentes para visualização dos dados no ambiente de Obras Raras (Figura 5.22) e Neutrinos Angra (Figura 5.23).

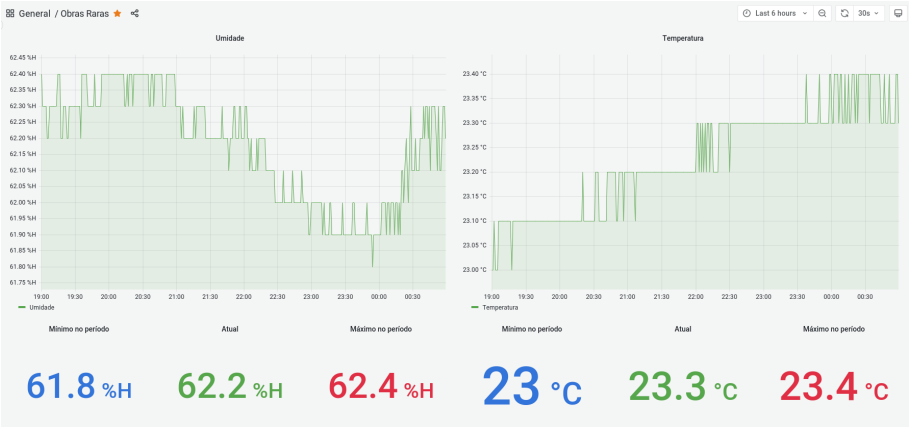


Figura 5.22: Visualização das variáveis monitoradas no ambiente de Obras Raras do CBPF.

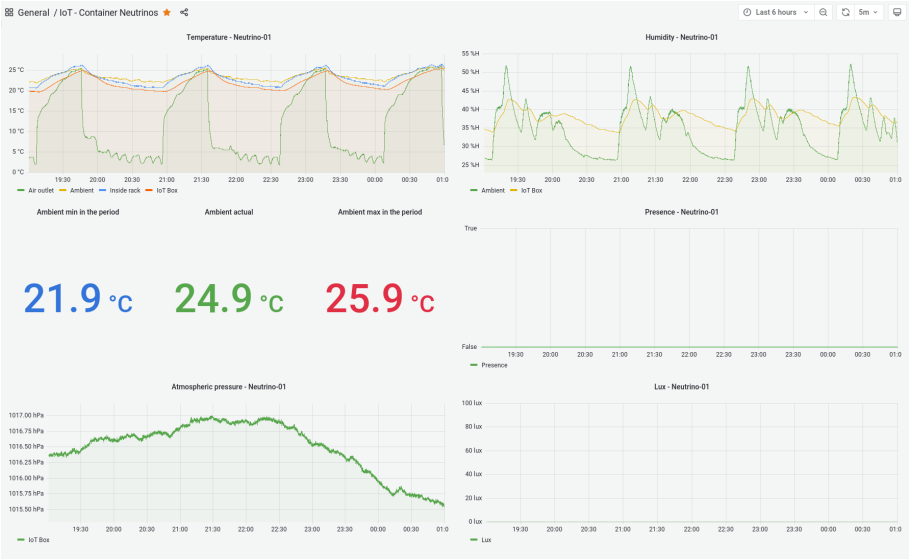


Figura 5.23: Visualização das variáveis monitoradas no laboratório contêiner.

A conclusão desse projeto também resultou na publicação de uma notícia no portal do CBPF [79].

5.8 Desafios na implementação

Durante a elaboração das bibliotecas (Apêndice A) foram realizados testes com os diversos tipos de sensores que estavam disponíveis para uso, com a finalidade de permitir a adaptação do projeto de forma rápida, de acordo com a necessidade de configuração. Os sensores testados foram: DS18B20 (temperatura); DHT11, DHT21, DHT22 (umidade e temperatura); Módulos GY49, TSL2561 e BH1750 (luminosidade); Módulos RCWL-0516 e PIR (presença); BMP280 (pressão atmosférica e temperatura); BME280 (pressão atmosférica, temperatura e umidade).

Nesse processo é comum aparecer problemas que não existiam quando os sensores são utilizados individualmente, muitas vezes devido a conflitos de *namespaces* por causa das bibliotecas utilizadas. Também ocorre do equipamento parar de funcionar devido a alguma atualização inesperada de uma biblioteca utilizada, levando a um processo exaustivo de busca de erros e correções.

Um problema complexo que foi identificado - e vale ser mencionado - aconteceu devido a incompatibilidade do sensor RCWL-0516 com a conexão Wi-Fi. O módulo com sensor RCWL-0516 é um detector de movimento baseado em micro-ondas, ótimo para humanos pois ignora os movimentos de outros objetos e pode detectar a presença através de alguns materiais não metálicos.

Inicialmente as medições feitas pelo sensor eram extremamente instáveis, apresentando oscilações de forma constante entre os estados de detecção e não detecção de presença. A área completamente preenchida no gráfico, representado na Figura 5.24, é devido a variação constante entre os estados de detecção de presença e não detecção do sensor.

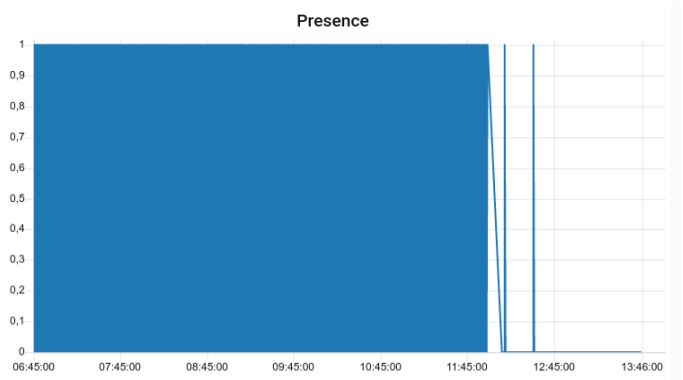


Figura 5.24: Resposta do sensor RCWL0516 com o Wi-Fi ligado durante a medição.

Após testes e consulta à documentação foi constatado que este sensor de micro-ondas trabalha na faixa dos 3,2GHz e a ativação da antena Wi-Fi do ESP32 é responsável por causar interferência [80]. Na verdade, é possível que qualquer antena Wi-Fi muito próxima seja capaz de causar interferência, então deve-se ter atenção com a localização e posicionamento deste sensor no ambiente.

A alternativa adotada foi programar a parte de utilização desse sensor de modo que a comunicação Wi-Fi fosse interrompida e a antena desligada antes da medição e, posteriormente, a comunicação fosse reestabelecida. Com isso o problema foi solucionado. No gráfico representado na Figura 5.25 é possível observar que a intermitência de detecção do sensor não ocorreu mais e foi possível fazer a integração com a biblioteca (Apêndice A).

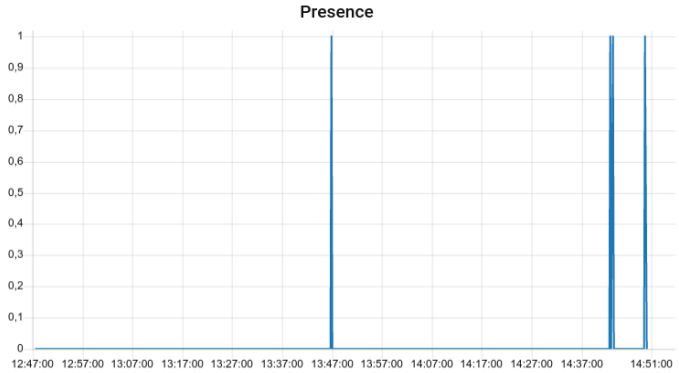


Figura 5.25: Resposta do sensor RCWL0516 com o Wi-Fi desligado durante a medição.

Outro desafio foi na utilização do GPS para sincronização da hora no Projeto Neutrinos. Como já foi discutido, o Raspberry Pi não possui um relógio interno para a hora. Quando o equipamento é reiniciado a sincronização é feita novamente pela rede. No caso da utilização do GPS essa etapa pode levar mais tempo, dependendo das condições da recepção do sinal. Para minimizar esse problema, o sistema só inicializa os processos de aquisição de dados após a sincronização da hora através do GPS.

Um problema identificado é que para que o detector de presença RCWL-0516 seja utilizado é necessário desligar a antena Wi-Fi durante a medição. Para que o sistema se mantenha responsivo é necessário que a antena seja desligada através de uma função de *callback* ativa. Com isso, caso a utilização desse sensor seja inevitável e necessite da resposta em tempo real, deve se optar pela comunicação através de um módulo para conexão cabeada. Uma outra possível alternativa é através de uma mudança de configuração do sistema e a utilização do protocolo ESP-NOW [81].

Para o caso de medições ambientais é necessário observar o local para disposição dos sensores de forma que as medições sejam realistas ao ambiente. Colocar um sensor de temperatura próximo a porta, por exemplo, pouco informará sobre as condições de temperatura dentro do laboratório. Caso o laboratório seja bastante frequentado, idealmente deve ser feita uma visita, com observação cuidadosa sobre a movimentação dos usuários, evitando colocar os sensores em locais com obstáculos ou escondidos.

Conforme estabelecido na Norma IEEE 2413-2019 [1], a platforma possui como característica a flexibilidade para rapidamente ser adaptada para a medição de outras variáveis ambientais. A biblioteca elaborada com o projeto, descrita no Apêndice A, está em contínuo desenvolvimento para a inclusão de novos sensores e correções de *bugs*.

Capítulo 6

Conclusão

O objetivo desta dissertação foi o desenvolvimento de uma plataforma para monitoramento que possa ser utilizada em laboratórios e experimentos. A implementação desse sistema foi dividida em seis partes:

- Análise dos requisitos;
- Montagem do equipamento;
- Implementação da arquitetura de comunicação;
- Desenvolvimento do *middleware*;
- Visualização dos dados e sistema supervisor;
- Segurança.

Sendo assim, este capítulo de conclusões será dividido nas seguintes seções: Análise dos requisitos; Montagem do equipamento; Implementação da arquitetura de comunicação; Desenvolvimento do *middleware*; Visualização dos dados e desenvolvimento do Supervisório; Segurança; Perspectivas futuras.

6.1 Análise dos Requisitos

Um dos maiores desafios desta etapa é conseguir identificar a idealização do sistema para os *stakeholders*. Com isso, uma análise de requisitos de qualquer projeto envolve um esforço com todos os *stakeholders*, desenvolvedores e usuários finais para alcançar um entendimento do que a solução proposta deve fazer. Isso sempre deve ser feito na fase inicial de qualquer projeto para garantir que a solução final desenvolvida esteja em conformidade com todos os requisitos.

6.2 Montagem do Equipamento

Os custos de produção para uma solução final também podem ser reduzidos através de otimizações no projeto envolvendo a produção de um circuito próprio, com a utilização dos componentes ao invés dos módulos.

Os módulos possuem como vantagem o fato de serem prontos para utilização, pois já incluem todos os dispositivos necessários para a utilização do componente.

6.3 Arquitetura de Comunicação

O CBPF possui uma excelente infraestrutura de redes sem fio e cabeada. Com isso a arquitetura de comunicação foi pensada de forma a atender às necessidades dos projetos, considerando a disponibilidade desses recursos. Caso a instalação seja feita em ambientes mais complexos ou restritos, o modo como o projeto foi elaborado facilita a adaptação, a exemplo do projeto Neutrinos Angra (Figura 5.7).

6.4 Desenvolvimento da API

A utilização do Python como linguagem base se mostrou bastante acertada, pois permitiu o desenvolvimento de todas as partes envolvendo a API, servidor, exibição dos dados e supervisório.

O Django também possui ferramentas que facilitam a implementação segura de API's, utilizando recursos de autenticação, como a biblioteca REST.

O trabalho utilizando API também facilita a manutenibilidade do projeto, permitindo a coexistência de versões diferentes de aplicação em operação. Essa flexibilidade é importante, pois permite que não seja necessária a implementação de uma possível atualização do *firmware* dos dispositivos de uma única vez.

A utilização de nomes grandes nos atributos facilita a leitura por humanos, mas também aumenta o *payload* para a comunicação. Como o sistema IoT privilegia a operação entre máquinas, é preferível nomear um atributo JSON como "v1" ao invés de "nome_do_meu_atributo", deixando a descrição como um comentário no código e na documentação do projeto.

6.5 Supervisório

O supervisório é um importante ambiente para o gerenciamento do sistema e permite o posterior desenvolvimento de diversas aplicações que poderão ser utilizadas até para automatizar o processo de coleta de dados de experimentos, bem como para a atuação, através do envio de comandos.

6.6 Segurança

O tema relacionado as boas práticas de segurança é amplo e nesta dissertação foi possível desenvolver uma abordagem que atende às necessidades de boa parte dos projetos que utilizam o ESP32. A parte de segurança é uma constante e, conforme estabelecido na norma IEEE 2413-2019 [1], é algo intrínseco ao desenvolvimento de sistemas IoT. Como já foi descrito no Capítulo 4, cada projeto deverá ser avaliado de acordo com as suas especificidades. As técnicas abordadas nesse trabalho não visam impedir os ataques, mas mitigar os possíveis efeitos causados em uma invasão.

No caso deste trabalho houve o cuidado de se habilitar somente o protocolo TLS v1.2 ou acima (o protocolo TLS v1.3 ainda não é suportado pela maioria das bibliotecas disponíveis para o ESP32) e em não utilizar versões vulneráveis dos algoritmos de criptografia [82] [83].

Caso seja necessária a utilização do protocolo TLS v1.3 é possível fazer a implementação utilizando o módulo criptográfico ATECC608A através da conexão I2C. Este chip já vem embutido no ESP32-WROOM-32SE e com ele é possível fazer todas as operações de criptografia mais modernas.

Outro aspecto é referente a conexão WiFi utilizada pelo ESP32. O ideal é possuir uma rede independente, utilizando o protocolo WPA2-PSK com AES (CCMP) e utilizar uma senha longa para dificultar os ataques de força-bruta [84], porém, mesmo isso não impede ataques mais sofisticados que exploram a vulnerabilidade do protocolo WPA2 [85].

Para a prevenção de ataques é essencial que também sejam implementadas formas físicas de proteção, através da utilização de sistemas de monitoramento ou blindagens para restringir o acesso ao equipamento. O constante monitoramento dos fluxos de comunicação entre o dispositivo e o servidor também é importante para identificação de alguma vulnerabilidade que possa surgir futuramente.

6.7 Perspectivas Futuras

O problema envolvendo o detector de presença baseado em micro-ondas é interessante e será investigado de forma mais profunda no futuro.

A comunicação através do ESP-NOW pode ser útil em ambientes mais restritos. Com isso cada ESP32 poderá funcionar como um nó independente de comunicação na rede, ampliando o espectro de possibilidade de projetos.

Com a facilidade trazida pelo projeto na aquisição dos dados, surge um problema: o acesso aos dados. Nos testes feitos utilizando as linguagens Python e R, começou

a ser significativo o tempo de acesso, seja através de conexão direta ao Banco de Dados ou manipulação de arquivos CSV. Com isso, se torna interessante utilizar tecnologias de Big Data, como o ecossistema Hadoop.

O sistema supervisor é uma das partes mais complexas do projeto, pois no futuro ele poderá ser transformado em um centro de integração de sistemas, onde existirá um portal que permitirá o gerenciamento e acesso de acordo com os requisitos de cada usuário/laboratório. Também está previsto o desenvolvimento de um *plugin* para o Grafana capaz de facilitar a implementação de *dashboards* para o projeto. Por fim, também está em fase de estudos a utilização da ferramenta ScadaBR [86] como supervisório alternativo da plataforma.

Ainda existe a ideia de implementar a parte de criptografia na comunicação utilizando o protocolo TLS v1.3, além da utilização de FPGA e *blockchain* [87], de forma a garantir a integridade dos dados.

Em resumo, está previsto para os projetos futuros:

- Investigar de forma mais profunda a causa da interferência com o sensor RCWL06;
- Implementar a comunicação entre os dispositivos através do protocolo ESP-NOW;
- Desenvolvimento de aplicações capazes de atuar no ambiente;
- Avaliação do limite de operação do sistema para um ambiente com muitos dispositivos conectados;
- Implementação de infraestrutura Big Data;
- Desenvolvimento e implementação de melhorias no supervisor;
- Integração da plataforma com o Grafana através de plugin;
- Integração com software de monitoramento de redes Nagios Zabbix através de SNMP;
- Integração da plataforma com o ScadaBR;
- Testes e inclusão de novos sensores;
- Suporte ao protocolo TLSv1.3.
- Utilização de módulo para conexão cabeada RJ45;
- Implementação de cadeia *blockchain* utilizando FPGA [88].

A diversidade de tecnologias envolvidas para a obtenção de uma plataforma foi um grande desafio nesse trabalho. O processo de desenvolvimento de soluções e inovação é bastante complexo. Transformar uma ideia em produto é algo desafiador.

Bibliografia

[1] IEEE. "Standard for an Architectural Framework for the Internet of Things (IOT) IEEE 2413", *IEEE-2413 Working Group. Technical Report*, 2019.

[2] "PZEM-004T Datasheet". https://innovatorsguru.com/wp-content/uploads/2019/06/PZEM-004T-V3.0-Datasheet-User-Manual.pdf, 2020. [Online; Dezembro-2021].

[3] "Split core current transformer". https://gavazziautomation.com/images/PIM/DATASHEET/ENG/CTA_DS_ENG.pdf, 2022. [Online; Junho-2022].

[4] "BME280 - Humidity sensor measuring relative humidity, barometric pressure and ambient temperature". https://www.bosch-sensortec.com/products/environmental-sensors/humidity-sensors-bme280/, 2021. [Online; Dezembro-2021].

[5] RAYES, A., SALAM, S. *Internet of things-from hype to reality: the road to digitization.* 2 ed. , Springer, 2019.

[6] ATZORI, L., IERA, A., MORABITO, G. "The internet of things: A survey", *Computer networks*, v. 54, n. 15, pp. 2787–2805, 2010.

[7] PIRES, P. F., DELICATO, F., BATISTA, T., et al. "Plataformas para a internet das coisas", *Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, 2015.

[8] "Internet das Coisas: um plano de ação para o Brasil". 2020. Disponível em: <https://www.bndes.gov.br/wps/portal/site/home/conhecimento/pesquisaedados/estudos/estudo-internet-das-coisas-iot/estudo-internet-das-coisas-um-plano-de-acao-para-o-brasil>.

[9] FURTADO, J., PINHEIRO, H., URIAS, E., et al. "Indústria 4.0: a quarta revolução industrial e os desafios para o Brasil", *A indústria do futuro no Brasil e no mundo. IEDI–Instituto de Estudos para o Desenvolvimento Industrial*, v. 3, pp. 171–195, 2019.

[10] "DECRETO No 9.854, DE 25 DE JUNHO DE 2019". https://www.in.gov.br/en/web/dou/-/decreto-n-9854-de-25-de-junho-de-2019-173021041, 2019. [Online; Novembro-2021].

[11] "Senado aprova isenções para incentivar Internet das Coisas; texto vai à sanção". https://www12.senado.leg.br/noticias/materias/2020/11/19/senado-aprova-isencoes-para-incentivar-internet-das-coisas-texto-vai-a-sancao, 2020. [Online; Novembro-2021].

[12] "Accelerating the Impact of IoT Technologies: SMEs & Industrial IoT". https://www.weforum.org/projects/accelerating-the-impact-of-iot-technologies, 2020. [Online; Outubro-2021].

[13] "Lei No 14.108, de 16 de dezembro de 2020". https://www12.senado.leg.br/noticias/materias/2020/11/19/senado-aprova-isencoes-para-incentivar-internet-das-coisas-texto-vai-a-sancao, 2020. [Online; Janeiro-2022].

[14] SCHWAB, K., DAVIS, N. *Aplicando a quarta revolução industrial*. Edipro, 2019.

[15] ALAHAKOON, D., YU, X. "Smart electricity meter data intelligence for future energy systems: A survey", *IEEE Transactions on Industrial Informatics*, v. 12, n. 1, pp. 425–436, 2015.

[16] TRAN, T. V., TAKAHASHI, H., NARABAYASHI, T., et al. "An Application of IoT for Conduct of Laboratory Experiment from Home". In: *2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, pp. 1–4, 2020. doi: 10.1109/IEMTRONICS51293.2020.9216375.

[17] PERKEL, J. M. *The Internet of Things comes to the lab*. Article 7639, Nature News, Washington, D.C., 2017.

[18] "TetraScience is the world's leading Life Sciences R&D Data Cloud." https://www.tetrascience.com/, 2020. [Online; Novembro-2021].

[19] POGHOSYAN, G., PEFKIANAKIS, I., LE GUYADEC, P., et al. "Extracting usage patterns of home IoT devices". In: *2017 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1318–1324. IEEE, 2017.

[20] "Hype Cycle for the Internet of Things, 2019". https://www.gartner.com/en/documents/3947474/hype-cycle-for-the-internet-of-things-2019, 2019. [Online; Janeiro-2022].

[21] MICHAL, V. *Security of IoT Devices Based on ESP32*. Dissertação de mestrado, Czech Technical University in Prague - Faculty of Information Technology, 2020. https://dspace.cvut.cz/handle/10467/89988/.

[22] "Lei No 11.934, de 05 de maio de 2009". https://www2.camara.leg.br/legin/fed/lei/2009/lei-11934-5-maio-2009-588054-norma-pl.html, 2009. [Online; Setembro-2021].

[23] NAKAGAWA, E. Y., OLIVEIRA ANTONINO, P., BECKER, M. "Reference architecture and product line architecture: A subtle but critical difference". In: *European conference on software architecture*, pp. 207–211. Springer, 2011.

[24] CLOUTIER, R., MULLER, G., VERMA, D., et al. "The concept of reference architectures", *Systems Engineering*, v. 13, n. 1, pp. 14–27, 2010.

[25] PIRES, P. F., CAVALCANTE, E., BARROS, T., et al. "A platform for integrating physical devices in the Internet of Things". In: *2014 12th IEEE International Conference on Embedded and Ubiquitous Computing*, pp. 234–241. IEEE, 2014.

[26] RAZZAQUE, M. A., MILOJEVIC-JEVRIC, M., PALADE, A., et al. "Middleware for internet of things: a survey", *IEEE Internet of things journal*, v. 3, n. 1, pp. 70–95, 2015.

[27] NAGY, M., KATASONOV, A., KHRIYENKO, O., et al. "Challenges of middleware for the internet of things". In: *Automation Control-Theory and Practice*, Intech, pp. 247–270, 2009.

[28] LI, W., SANTOS, I., DELICATO, F. C., et al. "System modelling and performance evaluation of a three-tier Cloud of Things", *Future Generation Computer Systems*, v. 70, pp. 104–125, 2017.

[29] SOMMERVILLE, I. *Engenharia De Software*. 10 ed. , PEARSON BRASIL, 2019.

[30] FRANCO. R, A. E. A. "Sistema IoT multipropósito para monitoramento de laboratórios e experimentos", *Notas Técnicas - CBPF*, v. 11, n. 2, pp. 27–37, 2021. Disponível em: <https://doi.org/10.7437/nt2236-7640/2021.02.004>.

[31] "ESP32 Series Datasheet". https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf, 2020. [Online; Novembro-2021].

[32] “Taiwan Semiconductor Manufacturing Company”. https://www.tsmc.com/english, 2022. [Online; Maio-2022].

[33] “PZEM004T”. https://github.com/olehs/PZEM004T, 2021. [Online; Dezembro-2021].

[34] “Patola PB-114”. https://www.patola.com.br/index.php?route=product/product&product_id=50, 2022. [Online; Junho-2022].

[35] “What is SELinux”. https://www.redhat.com/en/topics/linux/what-is-selinux, 2019. [Online; Dezembro-2021].

[36] “Mozilla SSL Configuration Generator”. https://ssl-config.mozilla.org/, 2021. [Online; Dezembro-2021].

[37] “Understanding middleware”. https://www.redhat.com/en/topics/middleware, 2022. [Online; Fevereiro-2022].

[38] “Django: The web framework for perfectionists with deadlines”. https://www.djangoproject.com/, 2005. [Online; Janeiro-2022].

[39] “Django REST framework”. https://www.django-rest-framework.org/, 2011. [Online; Janeiro-2022].

[40] FIELDING, R. T. *Architectural styles and the design of network-based software architectures*. University of California, Irvine, 2000.

[41] RUSSELL, B., VAN DUREN, D. *Practical internet of things security*. 2 ed. New York, Packt Publishing Ltd, 2012.

[42] “OWASP Internet of Things Project”. https://wiki.owasp.org/index.php/OWASP_Internet_of_Things_Project, 2018. [Online; Dezembro-2021].

[43] “IoT Device Criteria”. https://www.nist.gov/itl/executive-order-improving-nations-cybersecurity/iot-device-criteria, 2021. [Online; Dezembro-2021].

[44] “Executive Order on Improving the Nations’s Cybersecurity”. https://www.cisa.gov/executive-order-improving-nations-cybersecurity, 2022. [Online; Maio-2022].

[45] “CVE-2019-17391 - Fault Injection and eFuse protections”. https://nvd.nist.gov/vuln/detail/CVE-2019-17391, 2019. [Online; Dezembro-2021].

[46] “CVE-2019-15894 - Fault Injection and Secure Boot”. https://nvd.nist.gov/vuln/detail/CVE-2019-15894, 2019. [Online; Dezembro-2021].

[47] "CVE-2019-12587 - Zero PMK Installation". https://nvd.nist.gov/vuln/detail/CVE-2019-12587, 2019. [Online; Janeiro-2022].

[48] "CVE-2019-12586 - EAP DoS". https://nvd.nist.gov/vuln/detail/CVE-2019-12586, 2019. [Online; Dezembro-2021].

[49] "CVE-2018-18558 - Secure Boot Bypass". https://nvd.nist.gov/vuln/detail/CVE-2018-18558, 2018. [Online; Dezembro-2021].

[50] BUTUN, I., ÖSTERBERG, P., SONG, H. "Security of the Internet of Things: Vulnerabilities, attacks, and countermeasures", *IEEE Communications Surveys & Tutorials*, v. 22, n. 1, pp. 616–644, 2019.

[51] GENG, H. *Internet of things and data analytics handbook*. John Wiley & Sons, 2017.

[52] "HTTP vs. MQTT: A tale of two IoT protocols". https://cloud.google.com/blog/products/iot-devices/http-vs-mqtt-a-tale-of-two-iot-protocols, 2018. [Online; Dezembro-2021].

[53] "TCPdump". https://www.tcpdump.org/, 2021. [Online; Dezembro-2021].

[54] "Wireshark". https://www.wireshark.org/, 2021. [Online; Dezembro-2021].

[55] "ESPRESSIF - HTTPS server". https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/protocols/esp_https_server.html, 2021. [Online; Dezembro-2021].

[56] "Mbed TLS". https://github.com/ARMmbed/mbedtls, 2021. [Online; Dezembro-2021].

[57] "WiFiClientSecure". https://github.com/espressif/arduino-esp32/tree/master/libraries/WiFiClientSecure, 2021. [Online; Dezembro-2021].

[58] PRINETTO, P., ROASCIO, G. "Hardware Security, Vulnerabilities, and Attacks: A Comprehensive Taxonomy." In: *ITASEC*, pp. 177–189, 2020.

[59] TEHRANIPOOR, M., WANG, C. *Introduction to hardware security and trust*. Springer Science & Business Media, 2011.

[60] "The real story of STUXNET". https://spectrum.ieee.org/the-real-story-of-stuxnet, 2013. [Online; Dezembro-2021].

[61] RIZZOLO, R. F., FOOTE, T. G., CRAFTS, J. M., et al. "IBM System z9 eFUSE applications and methodology", *IBM Journal of Research and Development*, v. 51, n. 1.2, pp. 65–75, 2007.

[62] "ESPRESSIF Docs: Flash Encryption". https://docs.espressif.com/projects/esp-idf/en/latest/esp32/security/flash-encryption.html, 2018. [Online; Dezembro-2021].

[63] FRANCO. R, A. E. A. "Comunicações seguras entre dispositivos IoT utilizando o ESP32", *Notas Técnicas - CBPF*, v. 12, n. 2, pp. 25–30, 2022. Disponível em: <https://dx.doi.org/10.7437/NT2236-7640/2022.02.003>.

[64] "Apache JMeter". https://jmeter.apache.org/, 2022. [Online; Fevereiro-2022].

[65] DE NORMAS TÉCNICAS ABNT, A. B. "NBR 16401-2:2008 - Parâmetros de conforto térmico", 2008.

[66] "Grafana Labs". https://grafana.com/, 2022. [Online; Fevereiro-2022].

[67] "Zabbix". https://www.zabbix.com/, 2022. [Online; Fevereiro-2022].

[68] "Hunga Tonga-Hunga Ha'apai Erupts". https://earthobservatory.nasa.gov/images/149347/hunga-tonga-hunga-haapai-erupts, 2022. [Online; Fevereiro-2022].

[69] "Onda de choque do vulcão de Tonga percorre o mundo e chega ao Brasil." https://metsul.com/onda-de-choque-do-vulcao-de-tonga-percorre-o-mundo-e-chega-ao-brasil/, 2022. [Online; Fevereiro-2022].

[70] "Impactos da erupção do vulcão em Tonga foram detectados no laboratório do CBPF". https://www.gov.br/cbpf/pt-br/assuntos/noticias/impactos-da-erupcao-do-vulcao-em-tonga-foram-detectados-no-laboratorio-do-cbpf, 2022. [Online; Março-2022].

[71] DE MELLO, P. M. A. C., DA COSTA SANTOS, M. J. V., TEIXEIRA, R. P. A. "Manual de conservação de acervos bibliográficos da UFRJ", 2004.

[72] ANJOS, J., BARBOSA, A., BERNSTEIN, A., et al. "Angra dos Reis reactor neutrino oscillation experiment", *Brazilian Journal of Physics*, v. 36, n. 4A, pp. 1118–1123, 2006.

[73] GOMEZ GONZALEZ, L. F. *Estudo da relação sinal/ruído em detectores de antineutrinos de reator*. Tese de Doutorado, Universidade Estadual de Campinas, 2019.

[74] LIMA JR, H., ALFONZO, J., ANJOS, J., et al. "Neutrinos Angra experiment: commissioning and first operational measurements", *Journal of Instrumentation*, v. 14, n. 06, pp. P06010, 2019.

[75] DORNELAS, T. I. “Estudo, Desenvolvimento e Análise da Instrumentaçao do Experimento Neutrinos Angra”, 2015.

[76] “The Raspberry Pi Guide: For Scientists and anyone else.” https://raspberrypi-guide.github.io/raspberry-pi, 2022. [Online; Maio-2022].

[77] “NEO-M8 series.” https://www.u-blox.com/en/product/neo-m8-series, 2022. [Online; Maio-2022].

[78] LOMBARDI, M. A., NELSON, L. M., NOVICK, A. N., et al. “Time and frequency measurements using the global positioning system”, *Cal Lab: International Journal of Metrology*, v. 8, n. 3, pp. 26–33, 2001.

[79] “CBPF instala sistema de monitoramento IoT na Usina Nuclear Angra 2”. https://www.gov.br/cbpf/pt-br/assuntos-periodo-defeso/noticias/laboratorio-neutrinos-do-cbpf-instala-sistema-de-monitoramento-iot-na-usina-nuclear-ang 2022. [Online; Outubro-2022].

[80] “RCWL-0516 GitHub informations”. https://github.com/jdesbonnet/RCWL-0516, 2020. [Online; Outubro-2021].

[81] “ESP-NOW”. https://www.espressif.com/en/products/software/esp-now/overview, 2021. [Online; Dezembro-2021].

[82] MCKAY, K., COOPER, D. *Guidelines for the selection, configuration, and use of Transport Layer Security (TLS) implementations*. Relatório técnico, National Institute of Standards and Technology, 2017.

[83] “SSL/TLS Vulnerabilities”. https://www.hhs.gov/sites/default/files/securing-ssl-tls-in-healthcare-tlpwhite.pdf, 2021. [Online; Dezembro-2021].

[84] SALMON, A., LEVESQUE, W., MCLAFFERTY, M. *Applied Network Security*. Packt Publishing Ltd, 2017.

[85] VANHOEF, M., PIESSENS, F. “Key reinstallation attacks: Forcing nonce reuse in WPA2”. In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1313–1328, 2017.

[86] “ScadaBR - Software livre, gratuito e de código-fonte aberto, para desenvolvimento de aplicações de Automação, Aquisição de Dados e Controle Supervisório.” https://www.scadabr.com.br/, 2022. [Online; Maio-2022].

[87] XU, L., CHEN, L., GAO, Z., et al. "FPGA based blockchain system for industrial IoT". In: *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 876–883. IEEE, 2020.

[88] FLORIN, R., IONUT, R. "FPGA based architecture for securing IoT with blockchain". In: *2019 International Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, pp. 1–8, 2019. doi: 10.1109/SPED.2019.8906595.

Apêndice A

Diagramas

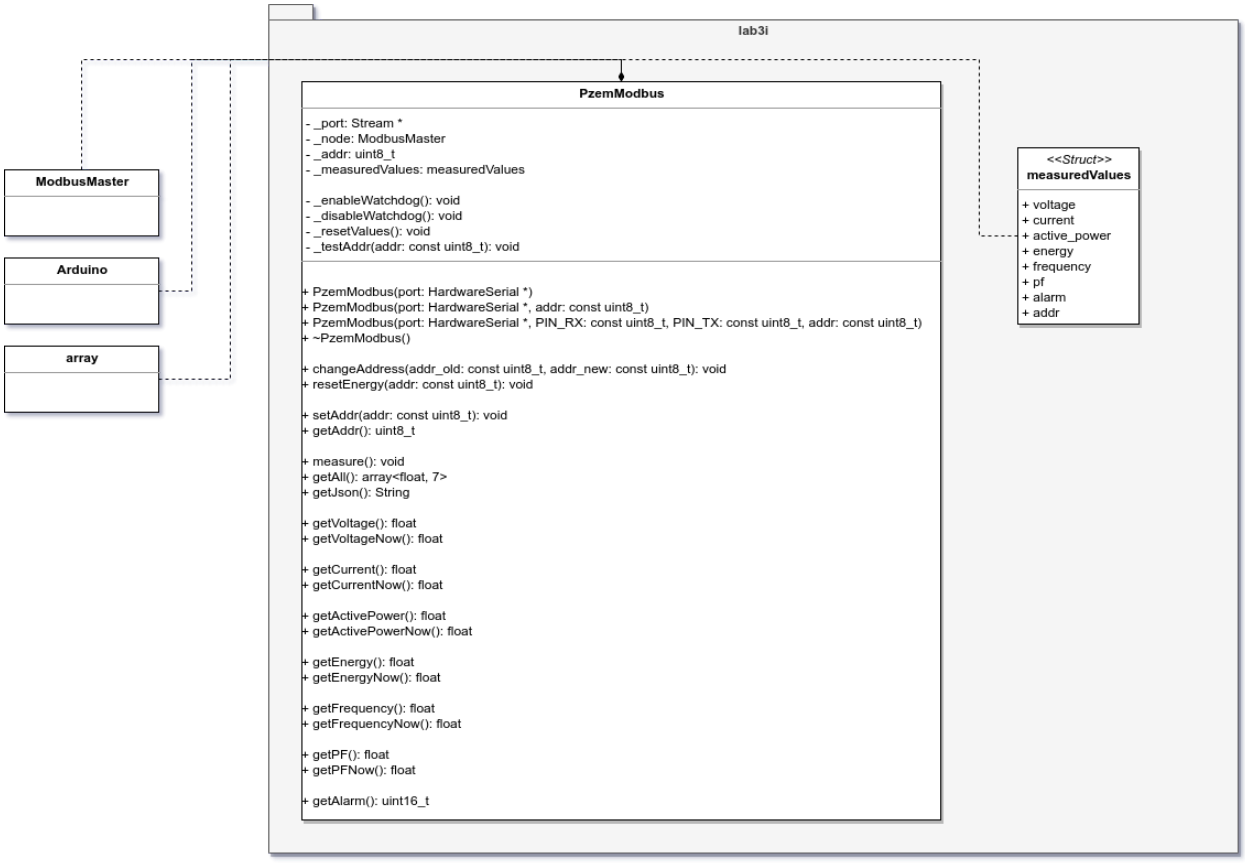


Figura A.1: Diagrama de Classe para o sensor PZEM

A bilioteca com o diagrama de classe representado na Figura A.2, atualmente suporta os seguintes sensores:

- Sensor de temperatura DS18B20;
- Sensores de temperatura e umidade DHT11, DHT21, DHT22, AM2301 e AM2302;
- Sensor de luminosidade GY49;

- Sensor de presença RCWL-0516;
- Sensor de pressão atmosférica e temperatura BMP280;
- Sensor de umidade, pressão atmosférica e temperatura BME280;
- Senso de luminosidade TLS2561.

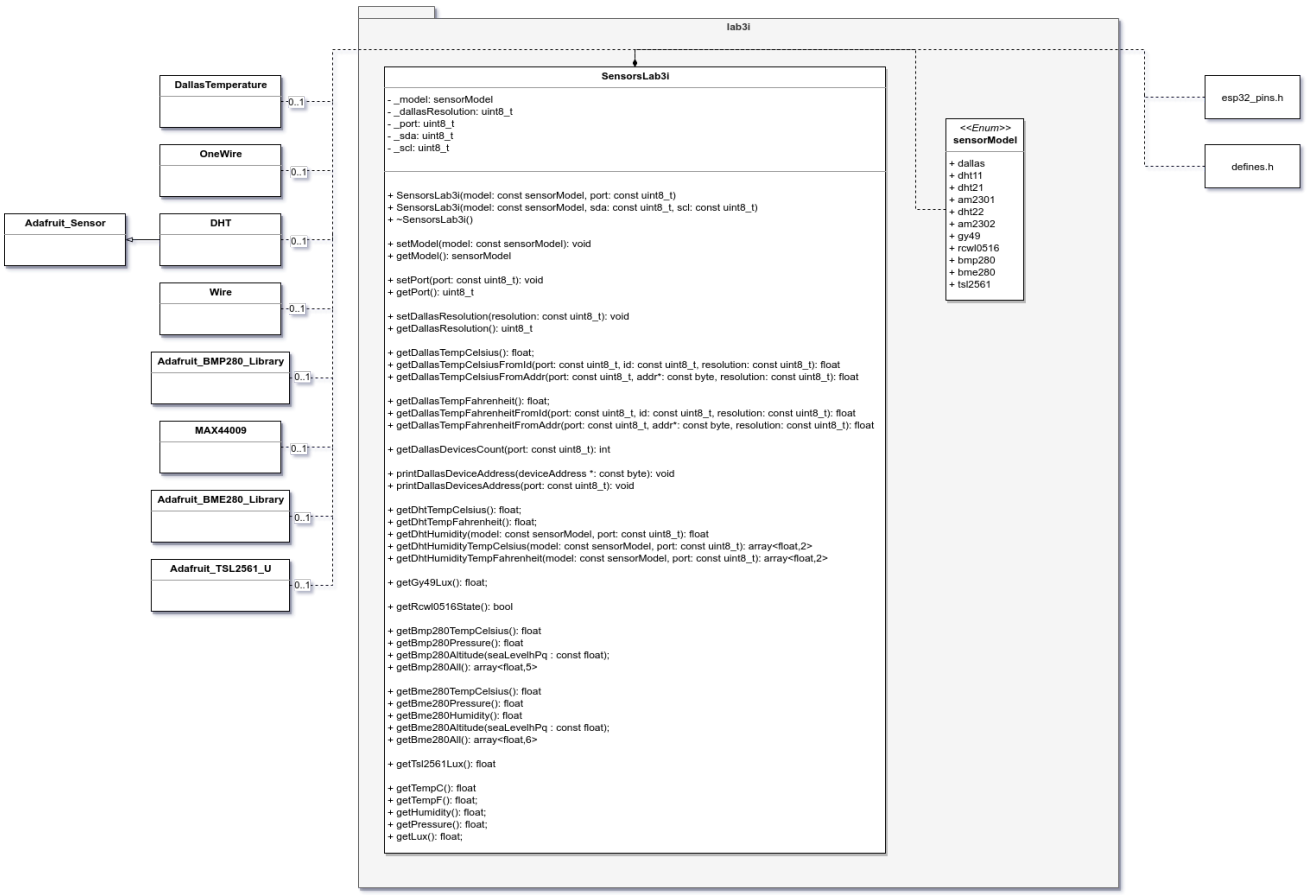


Figura A.2: Diagrama de Classe para o parser agregador de sensores

Para o futuro está previsto a integração com os sensores:

- Sensor de ozônio MQ-131;
- Sensor de gases inflamáveis MQ-2;
- Sensor de Monóxido de Carbono MQ-7;
- Sensor de hidrogênio MQ-8;
- Sensor de gases tóxicos MQ-135;

Apêndice B

Descrição da API

GET **/year-options/**

Retorna a lista com os anos que possuem medições disponíveis

Response application/json

200 ok

```
{
    "options":[
    2022,
    2021,
    2020
    ]
}
```

GET **/month-options/<year>/**

Retorna uma lista com os meses disponíveis para o ano selecionado

Parameter

year ano em que as medições foram feitas

Response application/json

200 ok

```
{
    "options":[
    1,
    2,
    3
    ]
}
```

GET **/day-options/<year>/<month>/**

Retorna uma lista com os dias disponíveis para o ano e mês selecionados

Parameter

year ano em que as medições foram feitas

month mês do ano em que as medições foram feitas

Response application/json

200 ok

```
{
    "options":[
    27,
    12,
    11,
    10
    ]
}
```

GET **/medidas-options/**

Retorna uma lista com todas as variáveis medidas disponíveis

Response application/json

200 ok

```
{
    "options":[
    "medida_id",
    "local",
    "sensor_name",
    "ip",
    "mac",
    "rede",
    "temperatura",
    "umidade",
    "pressao",
    "tensao_a",
    "tensao_b",
    "tensao_c",
    "corrente_a",
    "corrente_b",
    "corrente_c",
    "potencia_a",
    "potencia_b",
    "potencia_c",
    "energia_a",
    "energia_b",
    "energia_c",
    "frequencia_a",
    "frequencia_b",
    "frequencia_c",
    "fator_potencia_a",
    "fator_potencia_b",
    "fator_potencia_c",
    "data"
    ]
}
```

GET **/medidores-options/**

Retorna uma lista com todas as variáveis disponíveis dos medidores

Response application/json

200 ok

```
{
    "options":[
    "sensor_id",
    "local",
    "sensor_name",
    "ip",
    "mac",
    "rede",
    "versao",
    "compilado",
    "data"
    ]
}
```

POST **/medidores/**

Cria uma nova linha na tabela com os valores fornecidos.

Body application/json

```
{
    "local":"Lab3i",
    "ip":"10.0.0.5",
    "mac":"00:01:02:03:04:05",
    "rede":"RedeIOT",
    "versao":"3.2.1",
    "compilado":"06/10/2021"
}
```

POST **/medidas/**

Cria uma nova linha na tabela com os valores fornecidos.

Body application/json

```
{
    "local":"Lab3i",
    "sensor_name":"Sensor1",
    "ip":"10.10.6.23",
    "mac":"00:24:21:CC:FE:CC",,
    "rede":"CbpfIot",
    "temperatura":"23.2",
    "umidade":"70.00",
    "pressao":"1014.01",
    "tensao_a":"128.02",
    "tensao_b":"129.02",
    "tensao_c":"127.02",
    "corrente_a":"12.45",
    "corrente_b":"10.05",
    "corrente_c":"9.418",
    ...
}
```

Apêndice C

Produção Científica

Ao longo do desenvolvimento desta dissertação foram publicados duas notas técnicas e duas notícias relacionados ao projeto.

C.1 Notas Técnicas do CBPF

- Sistema IoT multipropósito para monitoramento de laboratórios e experimentos [30]: Utilizando os dispositivos ESP32 e Raspberry Pi, além dos programas Mosquitto e Node-RED, foi possível elaborar um sistema baseado em IoT capaz de fazer a aquisição de dados, obtidos através de sensores específicos, de variáveis ambientais (temperatura, umidade, pressão e luminosidade) e de energia elétrica (tensão, corrente, potência ativa, fator de potência e frequência). Além disso, o sistema ainda permite que seja feita a atuação remota para modificação do estado de funcionamento do equipamento.

- Comunicações seguras entre dispositivos IoT utilizando o ESP32 [63]: Neste trabalho foi desenvolvida uma metodologia para aumentar a segurança na comunicação em *Internet of Things* (IoT) que utilizem o ESP32. No projeto desenvolvido foram implementadas criptografias na camada de comunicação, utilizando o protocolo TLSv1.2, além da criptografia do *firmware*, para mitigar ataques físicos com o intuito de inserir um *firmware* modificado.

C.2 Notícias

- Impactos da erupção do vulcão em Tonga foram detectados no laboratório do CBPF [70].

- CBPF instala sistema de monitoramento IoT na Usina Nuclear Angra 2 [79].

Índice

A
Adaptação dinâmica, 10
agentes biológicos, 50
Agilidade, 5
AIEA, 51
Angra II, 51
Apache JMeter, 40
API, 25
API Key, 26
Application Programming Interface, 25
arquitetura de comunicação, 23
Arquitetura de Referência, 9

B
Banco Nacional de Desenvolvimento Econômico e Social, 2
Blockchain, 8
BME280, 14
BNDES, 2
boas práticas de segurança, 62

C
Camada de Dispositivos, 11
Camada de Nuvem, 11
Capacidade de compartilhamento, 5
CBPF, 49
Centro Brasileiro de Pesquisas Físicas, 4
chip NEO-M8N, 53
Ciência de contexto, 9
Cloud of Things, v, 11
coleta de dados, 3
Computação Ubíqua, 6
Criogenia, 43
criptografia, 62
critérios do projeto, 40
custos de produção, 61

D
dashboard, 55
Decreto N^o 9.854, 2
Descoberta e gerenciamento de dispositivos, 9
dispositivos IoT, 30
Django, 26
Django REST, 26

E
eFUSE, 36
Escalabilidade, 10
escalabilidade, 40
ESP32, 13

F
Fórum Econômico Mundial, 3

G
gateway, v
Gerenciamento de grandes volumes de dados, 10, 40
GPS, 53
Grafana, 47

I
IDS, 24

IEDI, 2
IEEE 2413-2019, 5, 27, 30, 38
Indústria 4.0, 3, 8
Instituto de Estudos para o Desenvolvimento Industrial, 2
Intelligent Operation Center, 27
Interface de alto nível, 9
Internet of Laboratory Things, 8
Internet of Things, v, 1
Interoperabilidade, 9

J
JSON, 26

L
Lei 14.108, 2

M
M2M, 2
Machine to Machine, 2
Man-in-the-middle, 31
manutenibilidade, 61
medidor de energia, 7
medidor inteligente, 7
medição elétrica, 7
Middleware, 2, 6, 9, 25
modo de monitoramento, 44
modo de operação autônomo, 17, 44
modo de operação de monitoramento, 18
modo de operação por mensagem, 18
Modularidade, 5
MQTT, 32
MVT, 26
Máquina a Máquina, 2

N
Network of Things, 1
Neutrinos Angra, 50, 51
NSA, 24
NTP, 53
Núcleo de Informação e Biblioteca, 45

O
onda de choque, 49
OWASP, 30

P
PL 6549/2019, 2
Plano Nacional de Internet das Coisas, 2
pressão atmosférica, 49
protocolo ESP-NOW, 59
protocolo NTP, 55
protocolo TLS, 62
PZCT-02, 14
PZEM-004T, 13

Q
quarta revolução industrial, 3

R
Radiação Cherenkov, 51
Real Time Clock, 53
responsivo, 59
resposta em tempo real, 59

S
ScadaBR, 63
Segurança, 10
segurança, 30
SELinux, 24
simulação de requisições, 40
stakeholders, 12
supervisor, 42

T
TLS, 24
topologia do ambiente, 23

V
viabilidade econômica, 22
Vulcão de Tonga, 49
vulnerabilidades, 30

W
Web Application Firewall, 24
WiFiClientSecure, 33

Z
Zabbix, 47