

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

FERNANDO FÁVERO

TIAGO VINICIUS BARROS PEREIRA

**ANÁLISE DOS PROTOCOLOS HTTP E MQTT PARA PROJETOS IOT
UTILIZANDO INTERNET FIBRA ÓPTICA E INTERNET MÓVEL**

PONTA GROSSA

2022

FERNANDO FÁVERO
TIAGO VINICIUS BARROS PEREIRA

**ANÁLISE DOS PROTOCOLOS HTTP E MQTT PARA PROJETOS IOT
UTILIZANDO INTERNET FIBRA ÓPTICA E INTERNET MÓVEL**

**Analysis of HTTP and MQTT Protocols for IOT Projects using Fiber optic
Internet and Mobile Internet**

Trabalho de Conclusão de Curso de apresentado como requisito para obtenção do título de Bacharel em Ciência da Computação, do Departamento Acadêmico de Informática, da Universidade Tecnológica Federal do Paraná (UTFPR).

Orientador: Prof. Dr. Augusto Foronda

PONTA GROSSA

2022



Esta licença permite compartilhamento, remixe, adaptação e criação a partir do trabalho, mesmo para fins comerciais, desde que sejam atribuídos créditos ao(s) autor(es). Conteúdos elaborados por terceiros, citados e referenciados nesta obra não são cobertos pela licença.

FERNANDO FÁVERO
TIAGO VINICIUS BARROS PEREIRA

**ANÁLISE DOS PROTOCOLOS HTTP E MQTT PARA PROJETOS IOT
UTILIZANDO INTERNET FIBRA ÓPTICA E INTERNET MÓVEL**

Trabalho de Conclusão de Curso apresentado como requisito para obtenção do título de Bacharel em Ciência da Computação, do Departamento Acadêmico de Informática, da Universidade Tecnológica Federal do Paraná (UTFPR).

Data de aprovação: 17 de novembro de 2022

Augusto Foronda
Doutorado
Universidade Tecnológica Federal do Paraná

Rogério Ranthum
Mestrado
Universidade Tecnológica Federal do Paraná

Luiz Rafael Schmitke
Doutorado
Universidade Tecnológica Federal do Paraná

PONTA GROSSA
2022

Dedicamos este trabalho às nossas famílias.

AGRADECIMENTOS

Ao professor orientador, Prof. Dr. Augusto Foronda, pelo conhecimento, suporte, paciência, incentivo e toda a orientação dada.

RESUMO

Com a constante evolução da tecnologia, nota-se cada vez mais o aumento de aparelhos conectados à internet, o que caracteriza o termo Internet das Coisas. Pensando na evolução da tecnologia, necessidade e desafios de cada projeto o desenvolvedor tem que adaptar a arquitetura IoT para que o funcionamento e a comunicação com o servidor seja a melhor possível, livre de erros e problemas para atingir o objetivo proposto. O responsável precisa saber qual o melhor meio que fará a comunicação com o servidor e ter conhecimento sobre os protocolos que serão utilizados. Este trabalho analisa a vazão de dados com os protocolos HTTP (*Hypertext Transfer Protocol*) e MQTT (*Message Queuing Telemetry Transport*) com diferentes meios de comunicação (Internet Fibra, Internet móvel 3G e 4G), através de um projeto de Internet das Coisas, que envia dados de temperatura e umidade relativa do ar para um servidor em nuvem e assim realizar uma análise entre os protocolos e meios de comunicação com os resultados obtidos. Os resultados mostram que o protocolo MQTT necessita menor banda e os meios de comunicação (Fibra, Internet móvel 3G e 4G) atendem a necessidade de banda do projeto.

Palavras-chave: Internet das Coisas; Protocolos de Comunicação; HTTP; MQTT.

ABSTRACT

With the constant evolution of technology, there is an increasing number of devices connected to the internet, which characterizes the term Internet of Things. Thinking about the evolution of technology, needs and challenges of each project, the developer has to adapt the IoT architecture so that the operation and communication with the server is the best possible, free of errors and problems to achieve the proposed goal. The person responsible needs to know which is the best way to communicate with the server and to have knowledge about the protocols that will be used. This work analyzes the data flow with the HTTP (Hypertext Transfer Protocol) and MQTT (Message Queuing Telemetry Transport) protocols with different means of communication (Fiber Internet, 3G and 4G mobile Internet), through an Internet of Things project, which sends data of temperature and relative humidity of the air to a cloud server and thus perform an analysis between the protocols and means of communication with the results obtained. The results show that the MQTT protocol requires less bandwidth and the means of communication (Fiber, 3G and 4G mobile Internet) meet the bandwidth needs of the project.

Keywords: Internet of Things; Communication Protocols; HTTP; MQTT.

LISTA DE ILUSTRAÇÕES

Figura 1 – Arquitetura IoT.....	17
Figura 2 – Comunicação Cliente/Servidor.....	19
Figura 3 – Formato de uma mensagem HTTP.....	20
Figura 4 – Requisição HTTP.....	21
Figura 5 – Formato de mensagem de resposta HTTP.....	21
Figura 6 – Resposta de uma requisição GET observada pelo Wireshark.....	21
Figura 7 – Funcionamento do protocolo MQTT.....	23
Figura 8 – Cabeçalho fixo da mensagem MQTT.....	24
Figura 9 – Cabeçalho fixo e payload da mensagem MQTT.....	24
Figura 10 – Raspberry PI.....	26
Figura 11 – Protótipo de IoT.....	28
Figura 12 – Topologia HTTP.....	29
Figura 13 – Canal <i>Thingspeak</i> com protocol HTTP.....	30
Figura 14 – Código para comunicação HTTP.....	30
Figura 15 – Execução do Código HTTP.....	31
Figura 16 – Topologia MQTT.....	31
Figura 17 – Canal <i>Thingspeak</i> com protocol MQTT.....	32
Figura 18 – Permissão para comunicação com o canal.....	32
Figura 19 – Código para comunicação MQTT.....	32
Figura 20 – Execução do Código MQTT.....	33
Figura 21 – Pacotes HTTP.....	34
Figura 22 – Pacotes MQTT.....	34

LISTA DE TABELAS

Tabela 1 – Média de tamanho dos pacotes HTTP	36
Tabela 2 – Média de tamanho dos pacotes MQTT	36
Tabela 3 – Comparação da vazão entre HTTP e MQTT	37
Tabela 4 – Comparação da vazão de protocolos HTTP com tempos de transmissão diferentes	37
Tabela 5 – Comparação da vazão de protocolos MQTT com tempos de transmissão diferentes	38
Tabela 6 – Comparação da vazão entre Internet Fibra, 4G e 3G.....	38
Tabela 7 – Média de latência das mensagens HTTP na rede Fibra	39
Tabela 8 – Média de latência das mensagens MQTT na rede Fibra	39
Tabela 9 – Média de latência das mensagens HTTP na rede 4G	39
Tabela 10 – Média de latência das mensagens MQTT na rede 4G	39
Tabela 11 – Média de latência das mensagens HTTP na rede 3G	40
Tabela 12 – Média de latência das mensagens MQTT na rede 3G	40

LISTA DE QUADROS

Quadro 1 – Métodos de requisição do HTTP	20
Quadro 2 – Tipos de mensagem vistas no Wireshark.....	24
Quadro 3 – Comparação entre os protocolos HTTP e MQTT.....	25
Quadro 4 – Velocidade de download e upload	27

LISTA DE ABREVIATURAS E SIGLAS

ACK	<i>Acknowledgment</i> - Reconhecimento
BPS	Bits por segundo
DUP	<i>Duplicate delivery</i> - Duplicação
HTTP	<i>Hypertext transfer protocol</i>
IOT	<i>Internet of things</i> – Internet das Coisas
IP	<i>Internet protocol</i> – Protocolo da internet
MQTT	<i>Message queuing telemetry transport</i>
QoS	<i>Quality of service</i> – Qualidade de serviço
TCP	<i>Transmission control protocol</i>
UDP	<i>User datagram protocol</i>

SUMÁRIO

1	INTRODUÇÃO	13
1.1	Definição do problema e motivação	14
1.2	Objetivos	14
1.2.1	Objetivo Geral.....	14
1.2.2	Objetivos Específicos	14
2	REFERENCIAL TEÓRICO.....	15
2.1	Internet das coisas	15
2.2	Arquitetura IoT.....	16
2.3	Protocolos de comunicação.....	17
2.3.1	Protocolo HTTP	18
2.3.2	Protocolo MQTT	22
2.4	Comparação entre os protocolos HTTP e MQTT	25
2.5	Raspberry Pi	25
2.6	Importância da rede em um projeto IoT.....	26
3	DESENVOLVIMENTO	28
3.1	Ambiente IoT.....	28
3.2	Protótipo IoT com protocolo HTTP	29
3.3	Protótipo IoT com protocolo MQTT	31
3.4	Visualização dos dados	33
3.5	Análise dos dados obtidos	34
3.5.1	Cenário 1	35
3.5.2	Cenário 2.....	37
3.5.3	Cenário 3.....	38
4	CONCLUSÃO	41
	REFERÊNCIAS.....	42

1 INTRODUÇÃO

O termo Internet das Coisas (IoT) refere-se a qualquer objeto físico que incorporado a sensores, softwares e outras tecnologias consiga conectar-se e trocar dados com outros dispositivos e sistemas pela internet (CASTRO, 2019). Vários componentes tornaram-se popularizados para projetos de automação IoT, entre os mais conhecidos e utilizados são o Arduino e o Raspberry. A arquitetura de comunicação em um projeto IoT é dividido por camadas: sensores, *gateway*, comunicação e servidores.

A camada inicial, composta por sensores, atua na captação das informações e passa os dados para a próxima camada, composta por uma placa microcontroladora, o Raspberry por exemplo, que é a camada *gateway*. Esta camada agrega os dados de entrada e saída e trabalha diretamente a comunicação com um serviço de gerenciamento, como por exemplo uma nuvem, onde os dados vão ser processados. Para a conexão entre a camada *gateway* e a camada de servidores (nuvem) são utilizados alguns protocolos (camada de comunicação), como por exemplo, o *Hypertext Transfer Protocol* (HTTP) e *Messaging Queue Telemetry Transport* (MQTT) (LEITE *et al.*, 2019).

O HTTP é um protocolo de comunicação bastante difundido em sistemas IoT, mas tem o problema de necessitar de uma alta largura de banda por conta da troca de mensagens entre o cliente e servidor (Soares, 2019). Já o MQTT, é um protocolo de transporte fim a fim e permite a comunicação entre entidade de um mesmo nível em sistemas finais, ou seja, clientes podem se comunicar no mesmo sistema. Esse protocolo foi projetado para ter uma capacidade computacional menor e baixa largura de banda, atendendo as necessidades de um projeto IoT, já que em alguns projetos não é possível ter uma conexão muito boa com o servidor (OLIVEIRA, 2017).

Várias tecnologias de conexão estão disponíveis para o envio de dados para a nuvem, como por exemplo, redes móveis e redes fibras ópticas. Cada rede tem suas características, como acessibilidade, flexibilidade e performance. Vários fatores influenciam para escolha ou não de determinado tipo de rede, como condições geográficas, custos de instalação ou manutenção, requisitos de performance ou aplicabilidade nas soluções que utilizam as mesmas (LUVISOTTO, 2016).

A conexão de fibra óptica possui velocidade de transmissão de dados podendo ser até um milhão de vezes maior que tecnologias antecessoras, como cabo

metálico e coaxial. Outro ponto positivo é sua estabilidade, uma prova disso é a sua utilização na conexão entre países através de cabos submarinos (TISOTT, 2021) .

Já as conexões móveis, sobretudo o 3G e 4G atualmente, são mais utilizadas nos aparelhos celulares, tablets, entre outros. Com essa tecnologia é possível ter praticidade no acesso à internet.

1.1 Definição do problema e motivação

Um projeto de IoT pode usar vários protocolos de comunicação para enviar as informações para a nuvem. É importante entender a diferença entre estes protocolos para poder definir qual é a melhor solução em determinado projeto. O protocolo escolhido tem relação direta com o meio de comunicação escolhido. Em uma conexão de fibra óptica tem-se uma largura de banda maior que uma conexão móvel.

Com o resultado deste trabalho será possível analisar se os dois protocolos estudados terão diferenças significativas quando testados em redes de fibra óptica e rede móvel.

1.2 Objetivos

Esta seção descreve os objetivos do trabalho. A seção 1.2.1 descreve o objetivo geral e na seção 1.2.2 são apresentados os objetivos específicos.

1.2.1 Objetivo Geral

O objetivo geral desse trabalho é analisar os protocolos HTTP e MQTT em um projeto IoT com conexão de internet fibra óptica e internet móvel.

1.2.2 Objetivos Específicos

- Desenvolver um protótipo de sistema IoT com o protocolo HTTP e MQTT.
- Definir métricas para comparar os protocolos estudados;
- Medir a vazão e atraso na transmissão de dados com o protocolo HTTP e MQTT em diferentes meios de conexão;
- Comparar os resultados.

2 REFERENCIAL TEÓRICO

Neste capítulo serão apresentados os conceitos básicos para a compreensão do trabalho e está dividido da seguinte forma: na seção 2.1 será apresentado a concepção de internet das coisas, assim como seu surgimento e a popularidade nos dias atuais. A seção 2.2 mostra a arquitetura de um projeto IoT. A seção 2.3 abordará os protocolos de comunicação que serão analisados neste trabalho, HTTP e MQTT. A seção 2.4 apresentará a placa Raspberry, assim como sua utilização em projetos para internet das coisas. Na seção 2.5 é abordada a importância da rede escolhida em um projeto IoT.

2.1 Internet das Coisas

A Internet das Coisas pode ser definida como “um novo mundo em que os objetos estarão conectados e passarão a realizar tarefas sem a interferência humana” (ASHTON, 2009). Segundo Ashton (2009), a falta de tempo das pessoas abre portas para que novas tecnologias surjam e que as atuais sejam adaptadas para fazer coisas que, de fato, não necessitam ser feitas por pessoas.

Para (REVELL, 2013) a Internet das Coisas refere-se à revolução em curso que pode ser observada no número crescente de dispositivos habilitados para internet. Nesse contexto, a IoT refere-se a um estado onde “coisas”, como objetos, ambientes, veículos estão capacitados a terem cada vez mais informações associadas a eles, e podem se conectar e se comunicar uns com os outros e com demais dispositivos habilitados para a web.

A conexão de dispositivos físicos com capacidade de detecção e comunicação (por meio de sensores e atuadores), não é um conceito novo, no entanto que refere à Internet das Coisas, os dispositivos físicos estão conectados através de endereços de IP exclusivos pelos quais dados podem ser reunidos e compartilhados. Como previamente citado, essa comunicação de dados acontece através de protocolos, sendo, por fim, armazenada na nuvem (CARRION; QUARESMA, 2019).

Paralelamente, o avanço das tecnologias de comunicação de redes de computadores foram se desenvolvendo e após a popularização do *Transmission Control Protocol/Internet Protocol* (TCP/IP) chegaram as redes WiFi, que garantiram a mobilidade de dispositivos, dispensando a utilização de fios para a conexão com a

internet. A comunicação com os dados se tornou ainda mais acessível a vários equipamentos e a qualquer local com a rede de telefonia celular 2G, 3G e 4G e a chegada recente do 5G.

A tecnologia do 5G vai permitir com sua velocidade de conexão cerca de 10 vezes mais rápida que sua antecessora (4G), com maior cobertura, com maior quantidade de equipamentos ligados e menor latência que, por exemplo, os carros autônomos possam se multiplicar pelas ruas se tornando mais comuns no nosso cotidiano. Além disso, o 5G proporcionará a evolução ainda maior da Realidade Virtual (RV), possibilitando o acompanhamento de cirurgias e até jogos em tempo real, *smart cities* (cidades inteligentes) e também na chamadas *smart factory*, que é a otimização de processos em um ambiente industrial.

2.2 Arquitetura IoT

De acordo com Avelar *et al.* (2010) a maioria dos projetos desenvolvidos utilizando do conceito de IoT fazem uso de arquiteturas de duas ou três camadas, as quais são: sensores, servidores e internet.

Nas duas arquiteturas, a camada de sensores é responsável pela coleta das informações do projeto. A camada de servidor tem como objetivo armazenar os dados coletados pela camada anterior. No caso da estrutura de 3 camadas, é utilizada a camada de internet como intermediário das duas, utilizando algum meio para que haja comunicação entre ambas (AVELAR *et al.*, 2010).

Para conseguir atender melhor os requisitos do projeto e tornar a proposta mais robusta, a camada de gateway foi incluída, sendo assim as camadas definidas foram as seguintes:

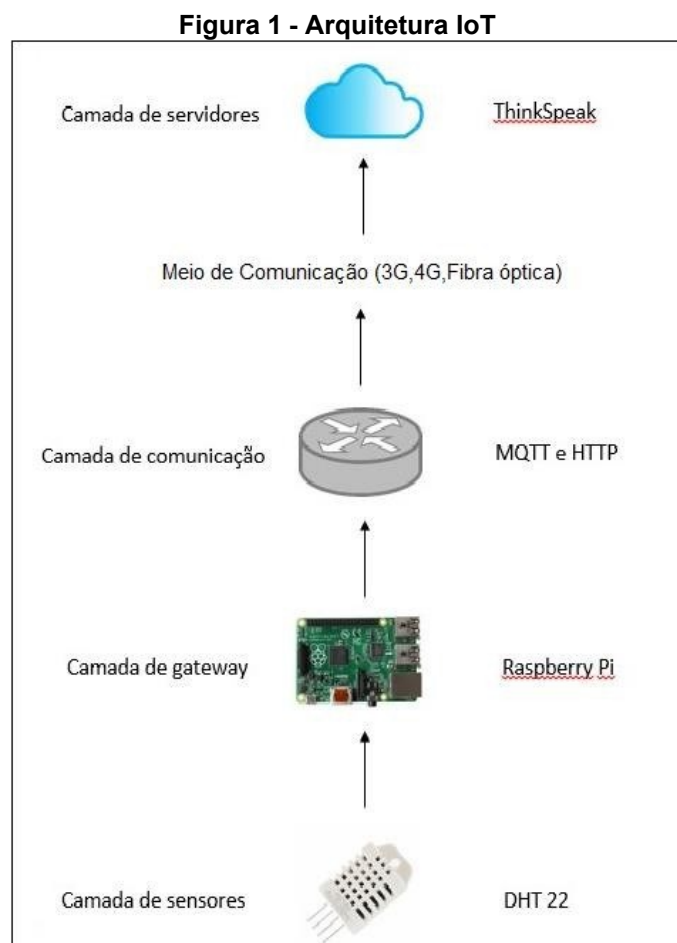
- Camada de sensores: onde está o sensor DHT 22, responsável pela coleta das informações, no caso deste trabalho, de temperatura e umidade;
- Camada de gateway: camada responsável por receber as informações da camada dos sensores e tratar os dados para que possam ser enviados para a camada dos servidores, sendo a camada que integra a placa Raspberry Pi;
- Camada de comunicação (Internet): trata-se da camada responsável da comunicação entre a camada de gateway e a camada de servidores. Nesse

projeto será utilizado a internet para a comunicação dessas camadas e os dois protocolos citados anteriormente: MQTT e HTTP;

- Camada de servidores: camada responsável pelo recebimento e principalmente armazenando e tratamento dos dados, neste projeto será utilizado a plataforma *ThingSpeak*.

E entre a camada de comunicação e servidores existem algumas opções como: fibra óptica e móvel.

A figura 1 ilustra como serão divididas as camadas explicadas anteriormente.



Fonte: Adaptado de Junior (2021)

2.3 Protocolos de comunicação

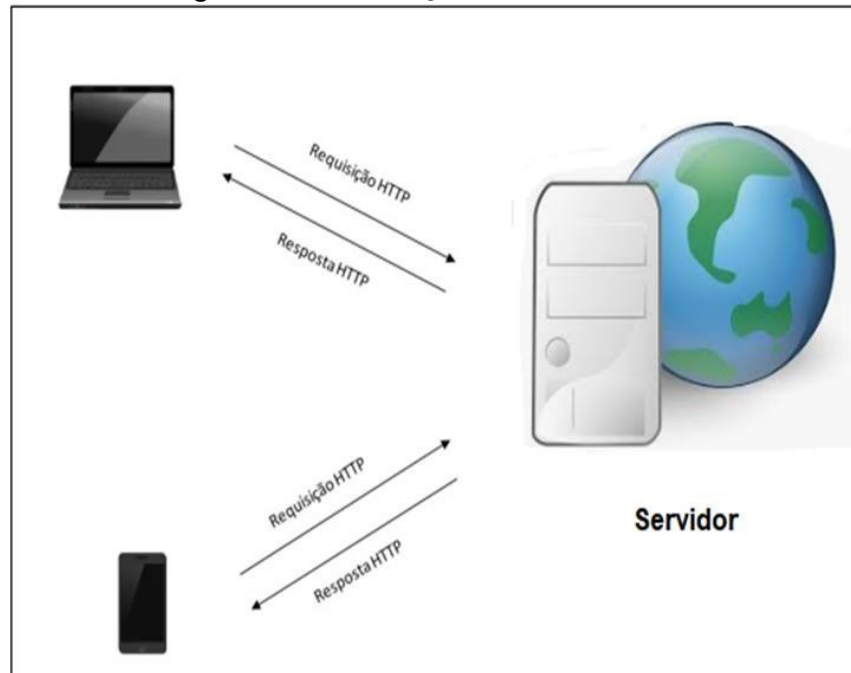
Os protocolos de comunicação são um conjunto de regras e procedimentos que tem como objetivo controlar a troca dos dados entre dispositivos, garantindo que essa troca ocorra de maneira eficiente e sem perdas. Para que isso ocorra, é necessário um padrão entre os dispositivos a partir do protocolo de comunicação.

O protocolo escolhido define os dados que farão parte da mensagem, o formato das mensagens, a estrutura dos pacotes que trafegarão na rede, os métodos para disseminar a informação e o endereçamento. Os principais elementos de um protocolo são sintaxe, semântica e temporização. A sintaxe diz respeito ao formato de dados e níveis de sinal, já a semântica refere-se as informações de controle para coordenação e tratamento de erros e por fim, a temporização trata dos aspectos de velocidade e sequência da informação.

2.3.1 Protocolo HTTP

O HTTP é um protocolo da camada de aplicação da internet. Segundo (KUROSE; ROSS, 2013) o HTTP está dividido em duas partes, o programa cliente e o programa servidor, os dois programas são executados em máquinas diferentes, conversam um com o outro por meio de mensagens HTTP. O HTTP define a estrutura destas mensagens e o modo como são trocadas entre o cliente e o servidor.

Segundo Quintanilla *et.al.* (2015), o protocolo HTTP é usado, principalmente, para acessar dados na Internet, permitindo a transferência de arquivos entre clientes e servidores, definindo uma linguagem de comunicação do tipo requisição-resposta, em que o cliente realiza uma solicitação ao servidor, que fica encarregado de retornar uma resposta, como mostra a Figura 2.

Figura 2 - Comunicação Cliente/Servidor

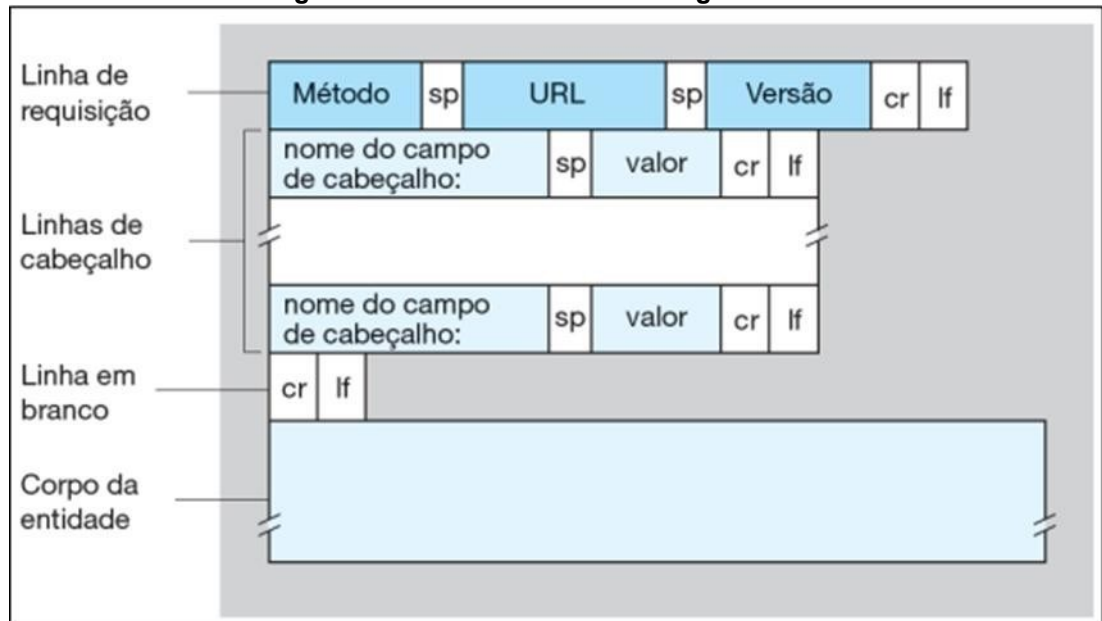
Fonte: Autoria Própria (2022)

Segundo (KUROSE; ROSS, 2013), existem 2 tipos de mensagem HTTP: de requisição e de resposta. De forma geral, a mensagem é composta por uma linha inicial, zero ou mais linhas de cabeçalho, uma linha em branco que indica o fim do cabeçalho, e o corpo da mensagem (SOUZA, 2015).

Tanto no tipo requisição e resposta, as linhas são escritas em texto ASCII comum, e cada linha é seguida de *'carriage return'* e *'line feed'* (CRLF), indicando o fim de cada linha. Para denotar a última linha (em branco) há um comando adicional de CRLF (KUROSE; ROSS, 2013).

A Figura 3 descreve o formato geral da mensagem de requisição do protocolo HTTP.

Figura 3 - Formato de uma mensagem HTTP



Fonte: Kurose e Ross (2013)

A primeira linha representa a linha de requisição contendo o método, a URL e a versão do protocolo HTTP. O Quadro 1 a seguir apresenta os quatro principais métodos de uma requisição HTTP.

Quadro 1 – Métodos de requisição do HTTP

Método	Ação
GET	Solicita algum recurso do servidor, como por exemplo, informações de um produto, imagens, etc...
POST	Envia dados específicos para serem processados no servidor, como por exemplo, informações de um formulário HTML
PUT	Envia alguma informação para o servidor, como por exemplo, atualização de um número de telefone
DELETE	Envia uma solicitação de exclusão para o servidor

Fonte: Adaptado da RFC 9110 (2022)

A Figura 4 apresenta um exemplo de uma mensagem de requisição HTTP, obtida através do software *Wireshark*.

Figura 4 - Requisição HTTP

```

Hypertext Transfer Protocol
>HTTP/1.1 301 Moved Permanently\r\n
Date: Sun, 26 Apr 2015 16:30:50 GMT\r\n
Server: Apache\r\n
Location: http://www.cert.br/\r\n
Keep-Alive: timeout=15, max=100\r\n
Connection: Keep-Alive\r\n
Transfer-Encoding: chunked\r\n
Content-Type: text/html; charset=iso-8859-1\r\n
\r\n
[HTTP response 1/1]
[Time since request: 0.039427000 seconds]

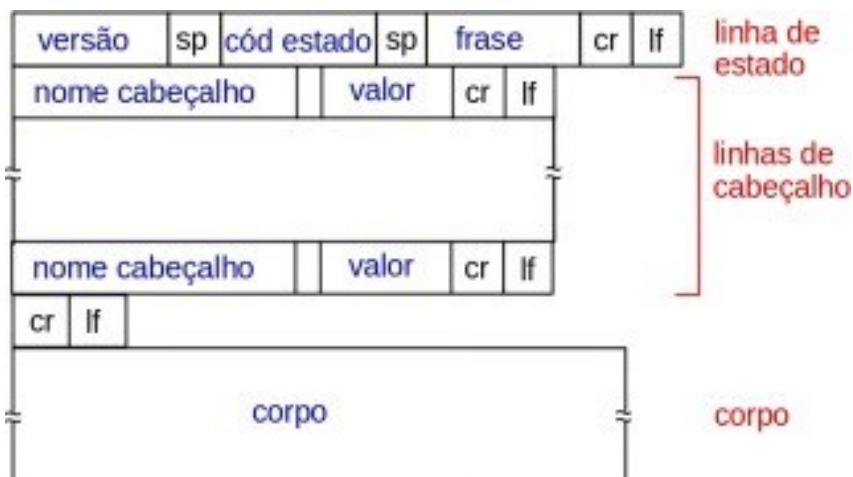
```

Fonte: Quintanilla et al., 2015

A mensagem de resposta é dividida em três seções: linha de estado, linhas de cabeçalhos e, por último, o corpo da entidade. A linha de estado indica a versão do HTTP, código de estado e a respectiva mensagem do estado. As linhas de cabeçalho denotam o tipo de conexão, data e hora da resposta enviada pelo servidor, qual servidor foi utilizado, data e hora da última modificação, tamanho em bytes do objeto a ser enviado e, por último, o tipo do conteúdo (QUINTANILLA *et al.*, 2015).

O formato de mensagem de resposta HTTP pode ser visto na Figura 5.

Figura 5 - Formato de mensagem de resposta HTTP



Fonte: Kurose e Ross (2013)

Um exemplo de resposta obtida pelo *Wireshark* é mostrado na Figura 6.

Figura 6 - Resposta de uma requisição GET observada pelo Wireshark

```

#Hypertext Transfer Protocol
↳HTTP/1.1 301 Moved Permanently\r\n
Date: Sun, 26 Apr 2015 16:30:50 GMT\r\n
Server: Apache\r\n
Location: http://www.cert.br/\r\n
Keep-Alive: timeout=15, max=100\r\n
Connection: Keep-Alive\r\n
Transfer-Encoding: chunked\r\n
Content-Type: text/html; charset=iso-8859-1\r\n
\r\n
[HTTP response 1/1]
[Time since request: 0.039427000 seconds]

```

Fonte: Quintanilla *et al.*, 2015

As mensagens de resposta do HTTP são divididas em cinco categorias, cada categoria possui um código específico de três números, sendo o primeiro deles fixo de acordo com a sua categoria. A seguir estão listadas as categorias e o que representam:

- Código 1xx: representa uma mensagem do tipo informação. Ex: código 100 significa que o servidor aceitou a requisição do cliente;
- Código 2xx: representa uma mensagem de sucesso. Ex: código 200 significa que a requisição feita ao servidor foi bem sucedida;
- Código 3xx: representa uma mensagem de redirecionamento. Ex: código 301 significa que a página foi movida;
- Código 4xx: representa uma mensagem de erro no cliente. Ex: código 404, página não encontrada;
- Código 5xx: representa uma mensagem de erro no servidor. Ex: código 500, erro interno no servidor.

2.3.2 Protocolo MQTT

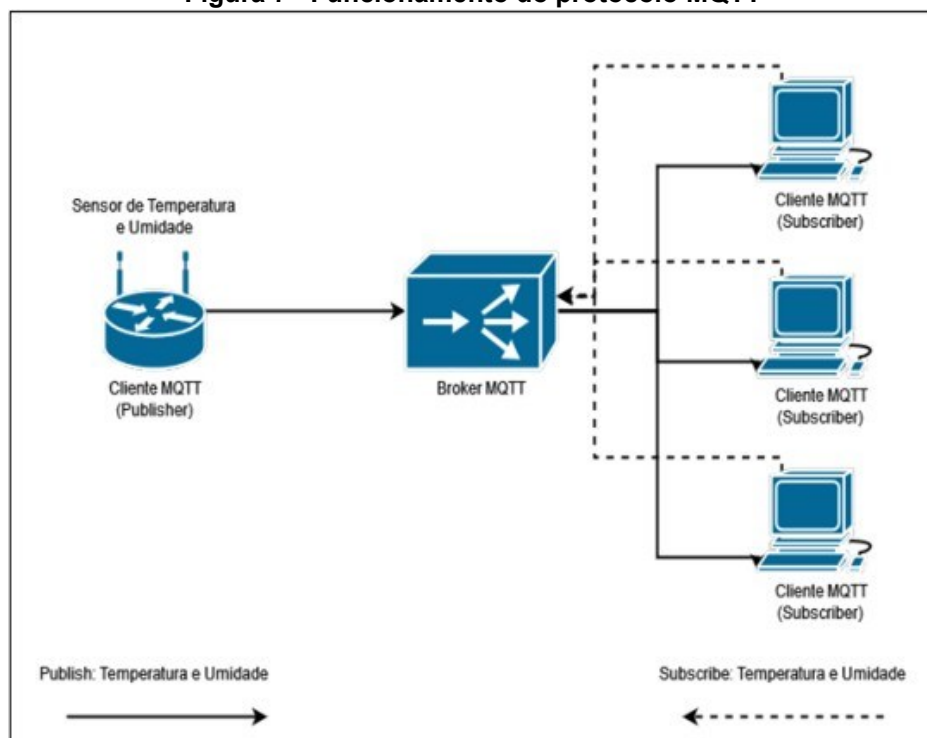
O MQTT, criado em 1999 por Andy Stanford-Clark e Arlen Nipper, é um protocolo de mensagens baseado na arquitetura *publisher/subscriber*, para uso em dispositivos restritos e redes não confiáveis. Os princípios do protocolo são minimizar os requerimentos de hardware de dispositivo e de largura de banda para garantir confiabilidade e garantia na comunicação (JUNIOR, 2021). A especificação do MQTT apresenta as características do protocolo, abaixo estão listadas algumas:

- Eficiente em termos de largura de banda: É possível enviar um quadro com apenas 2 bytes;
- Uso do TCP para fornecer conectividade;

- Oferece níveis de qualidade de serviço com três níveis de qualidade;
- Mecanismo que notifica partes interessadas quando um cliente se desconecta da rede anormalmente.

O MQTT é formado por três componentes: *subscriber*, *publisher* e o *broker*. Os clientes (*subscribers*) se conectam em determinado tópico de um *broker* para receber os dados de interesse que serão enviados por publicadores (*publishers*) até o *broker* (YASSEIN, 2017). Na Figura 7, é possível notar que o cliente MQTT à esquerda é um sensor que publica informações de temperatura e umidade relativa. Já na direita, trata-se de um cliente MQTT que é *subscriber* das informações geradas pelo sensor.

Figura 7 - Funcionamento do protocolo MQTT



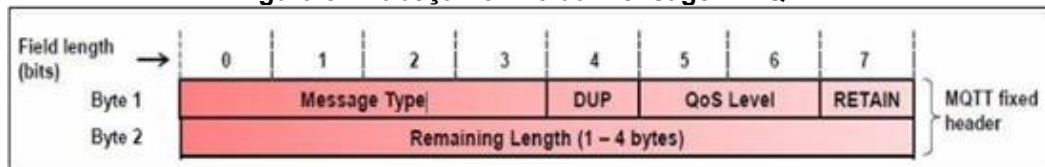
Fonte: MUNHOZ (2022)

Uma mensagem MQTT possui um cabeçalho fixo de 2 bytes, como mostra a Figura 7, sendo o primeiro byte, do bit 0 ao 3 serve para identificar o tipo da mensagem. O bit 4 representa o *Duplicate delivery* (DUP), que se trata do identificador de mensagem duplicada.

Os bits 5 e 6 identificam a Qualidade de Serviço (QoS), que indica o nível de garantia da entrega de uma mensagem *publish*. Por último, o bit 7 representa o RETAIN, que com o valor ativado faz com que o servidor retenha a mensagem *publish*

mesmo após ela ser entregue aos *subscribers* e que em um próximo evento de subscrição de um tópico, essa mensagem retida seja enviada para o novo assinante.

Figura 8 - Cabeçalho fixo da mensagem MQTT

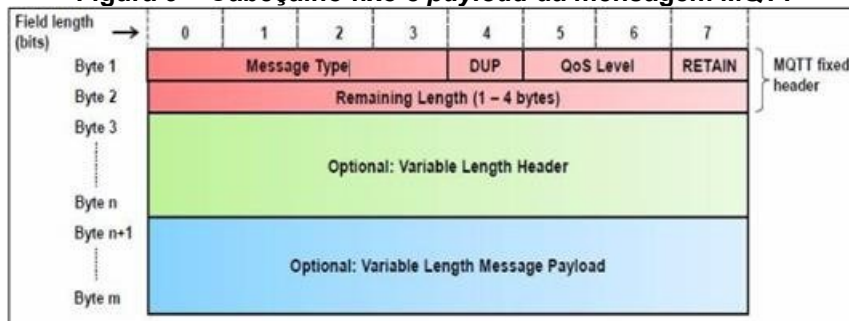


Fonte: Senin, Bussy (2022)

A largura restante do cabeçalho fixo (byte 2) é usada para representar a quantidade de bytes remanescentes na mensagem. Incluindo dados do cabeçalho variável e do *payload*. Determinados de mensagem MQTT apresentam cabeçalho variável, como mostra a Figura 9, ele fica localizado entre o cabeçalho fixo e o *payload*.

Usado principalmente nas mensagens do tipo CONNECT, este cabeçalho possui dois campos para nome e versão do protocolo, respectivamente e mais uma série de marcadores que definirão algumas diretivas para a conexão entre cliente e servidor.

Figura 9 – Cabeçalho fixo e *payload* da mensagem MQTT



Fonte: Senin, Bussy (2022)

O Quadro 2 apresenta, resumidamente, quatro tipos de mensagem MQTT vistas no *Wireshark*.

Quadro 2 - Tipos de mensagem vistas no Wireshark

CONNECT	CONNACK	PUBLISH	DISCONNECT
Requisição de conexão do cliente ao servidor	ACK de conexão	Publicação da mensagem	Cliente desconectado

Fonte: Autoria Própria (2022)

O MQTT tem suporte a três níveis de qualidade de serviço segundo IBM & Eurotech (2010), sendo eles:

- Valor 0: onde a mensagem é entregue uma vez ou sequer chega ao destino;
- Valor 1: onde a mensagem é reenviada até receber uma confirmação, mesmo que ocasione redundância;
- Valor 2: onde a mensagem é sempre entregue uma vez, tornando-se o modo mais confiável apesar de mais lento.

2.4 Comparação entre os protocolos HTTP e MQTT

Por mais que os dois protocolos tenham a mesma finalidade, transportar uma mensagem, eles utilizam diferentes formas e arquiteturas para isso. O Quadro 3 mostra as informações de cada protocolo e apresenta a diferença para cada categoria informada.

Quadro 3 - Comparação entre os protocolos MQTT e HTTP

Categorias	MQTT	HTTP
Modelo de comunicação	Funciona no modelo de publicação / assinatura	Funciona no modelo solicitação / resposta
Complexidade	Pequena	Grande
Protocolo da camada inferior	TCP	TCP e UDP
Tamanho da mensagem	Tamanho da mensagem é menor, pois usa formato binário	O tamanho da mensagem gerada é maior porque usa o formato ASCII
Tamanho do cabeçalho	2 bytes	8 bytes
Níveis de serviço	3 níveis (<i>broker, publish</i> e <i>subscriber</i>)	Todas as mensagens recebem o mesmo nível de serviço
Distribuição dos dados	Pode ser 1 para 1, 1 para nenhum ou 1 para muitos	Somente 1 para 1
Indicado para qual situação	Projeto para dispositivos de rede com recursos limitados, como IoT/M2M	Projetos que necessitam de um número maior de dados transportados

Fonte: Adaptado de Craggs (2022)

2.5 Raspberry Pi

O projeto Raspberry Pi, visto na Figura 10, é um projeto de hardware aberto que utiliza um processador da família ARM. A sua versão 3, tem um processador de 1,2 GHz de 64 bits, além de contar com WiFi, Bluetooth, um gigabyte de memória

RAM, quatro portas USB, 40 pinos de entrada e saída, saída HDMI e diversas outras interfaces, como câmera e memória externa (MONK, 2016).

Figura 10 – Raspberry PI



Fonte: Página do Raspberry PI (2022)

Inicialmente, o Raspberry Pi foi proposto como uma solução de um computador popular, capaz de executar uma versão do sistema operacional Linux, com interface gráfica e pacotes de aplicativos de código aberto. Porém, o sucesso do modelo fez com que o dispositivo se torna-se útil em várias outras aplicações, especialmente em sistemas embarcados. Suas interfaces de entrada e saída, bem como seu tamanho reduzido e as demais interfaces, facilitam vários tipos de aplicações (JUNIOR, 2021).

Por seu desempenho, é a placa mais utilizada para projetos IoT de pequena a grande complexidade, como por exemplo, automatizar casas através de módulo e sensores conectados a placa e também automatizar processos da agricultura.

2.6 Importância da rede em um projeto IoT

A necessidade das pessoas de estarem conectadas à internet, em qualquer lugar que estejam e a qualquer hora do dia, seja para trabalhar, lazer ou outra coisa, seja por celular, tablet, computador, etc é um fator de incentivo para modernidade, onde as operadoras estão investindo cada vez mais para obter um melhor compartilhamento de rede móvel no mercado (Alecrim, 2022).

Em um projeto IoT, para que o mesmo funcione de forma correta, é essencial que os tempos entre comandos e respostas, seja o menor possível. Nesse cenário

dois conceitos são muito relevantes na escolha de qual conexão utilizar: a latência e a velocidade de dados.

A latência nada mais é que o tempo de resposta entre uma ação e sua reação. A velocidade é inversamente proporcional a latência, isso é, quanto menor a latência, maior a velocidade de transmissão. Logo, para que a latência seja baixa, a velocidade deve ser alta.

A eficiência do protocolo de comunicação, no caso desse trabalho HTTP e MQTT, visando à parte de economia de recursos, em um projeto IoT pode ser verificada a partir do cálculo de vazão dos dados. A ideia principal é que quanto menor o consumo de banda da internet, melhor o protocolo será. Essa questão tem bastante impacto principalmente em projetos onde o recurso disponível é limitado. Pode-se pensar no caso específico de projetos IoT que usam a conexão de internet móvel, já que muitos planos de internet móvel possuem limitações de consumo.

As redes móveis 3G e 4G, que são objeto de estudo desse trabalho, possuem velocidades distintas, como pode ser visto no Quadro 4.

Quadro 4 - Velocidade de *download* e *upload*

Rede	Velocidade média de <i>download</i>	Velocidade média de <i>upload</i>
3G – Padrão WCDMA	2,0 Mbps	475 Kbps
4G	15,06 Mbps	7,63 Mbps

Fonte: TELECO (2020)

Uma dos pontos positivos das redes móveis é a questão de cobertura do sinal. Segundo dados de Agosto de 2022, disponibilizados na internet pelo site da Teleco, 5537 municípios (valor que representa 99,4% dos municípios do Brasil) possuem cobertura de internet 3G, sendo disponibilizadas pelas principais operadoras de telefonia do país (Vivo, Claro, Tim, etc). Já para a tecnologia do 4G, a cobertura contempla 5468 municípios, valor que representa 98,16 % dos municípios do país.

3 DESENVOLVIMENTO

Neste capítulo será apresentado a preparação do ambiente de comunicação entre o Raspberry e o servidor *ThingSpeak* utilizando os protocolos HTTP e MQTT. Assim como a medição da vazão e o tempo de resposta de ambos protocolos para os diferentes tipos de rede.

Para uma melhor organização do projeto, os testes foram separados em três cenários: primeiro, a comparação do MQTT com o HTTP no intervalo de tempo de 10 segundos em uma internet fibra; segundo, comparação em diferentes intervalos de tempo, e por último, comparação de ambos com internet fibra e móvel.

3.1 Ambiente IoT

O protótipo IoT foi montado, como mostra a Figura 11, a partir da conexão do módulo DHT 22 ao Raspberry pelas portas de alimentação, 01 DC Power e 06 Ground, e a porta para dados foi definida a 07 GPIO 4.

O DHT22 é um sensor de temperatura e umidade de baixo custo e interface serial a um fio. Com ele pode fazer leituras de temperaturas entre -40 a +80 graus Celsius e umidade entre 0 a 100%, tendo uma precisão de 0,5 grau para a temperatura e 2% para a umidade relativa do ar.

Figura 11 – Protótipo de IoT



Fonte: Autoria Própria (2022)

Para desenvolvimento do algoritmo de comunicação foi utilizado a linguagem de programação Python e algumas bibliotecas precisaram ser utilizadas. A `Adafruit_DHT` é uma biblioteca que permite a comunicação com o módulo de umidade

e temperatura, já `httplib` e `paho.mqtt` foram necessárias para comunicação com o servidor.

3.2 Protótipo IoT com protocolo HTTP

A topologia desse projeto com o protocolo HTTP consiste na coleta das informações do sensor DHT 22 pelo Raspberry PI e o envio das informações através de uma requisição POST para o servidor *ThingSpeak* e enviando como resposta o código de sucesso. O celular envia a requisição com o método GET para o servidor e recebe como resposta os dados de temperatura e umidade, como mostra a Figura 12.

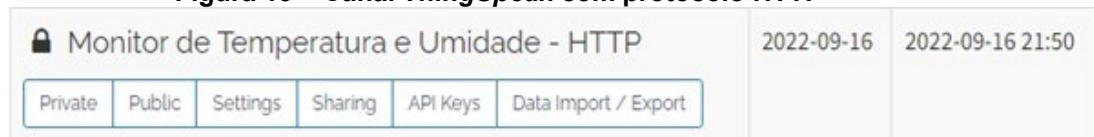
Figura 12 – Topologia HTTP



Fonte: Autoria Própria (2022)

Para realização do projeto IoT foi criado um canal no servidor *ThingSpeak* com o nome Monitor de Temperatura e Umidade – HTTP, como mostrado na Figura 13. Para a comunicação e envio dos dados foi desenvolvido um algoritmo, mostrado na Figura 14.

Figura 13 – Canal *ThingSpeak* com protocolo HTTP



Fonte: Autoria Própria (2022)

Figura 14 – Código para comunicação HTTP

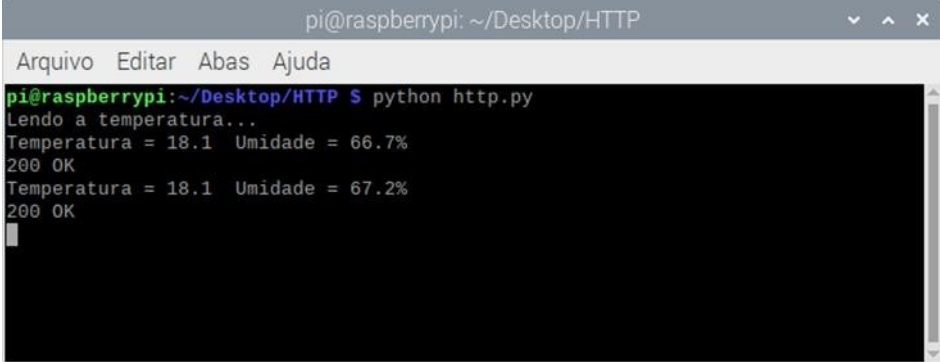
```

1 # Carrega as bibliotecas
2 import urllib
3 import urllib
4 import thingspeak
5 import Adafruit_DHT
6 import RPi.GPIO as GPIO
7 import time
8
9 #dados do canal do Thingspeak
10 channel_id = 1866036
11 write_key = 'LN250SAE00NYA6HW'
12 read_key = '1K3878JLM13ZF2X5'
13
14 # Define o tipo de sensor = Adafruit_DHT.DHT22
15 sensor = Adafruit_DHT.DHT22
16 GPIO.setmode(GPIO.BOARD)
17
18 # Define a GPIO conectada ao pino de dados do sensor (pino 16)
19 gpio_sensor = 4
20
21 print ("Lendo a temperatura...");
22 while(1):
23     # Efetua a leitura do sensor
24     umidade, temperatura = Adafruit_DHT.read_retry(sensor, gpio_sensor)
25     print ("Temperatura = {0:0.01f} Umidade = {1:0.01f}%").format(temp, umid)
26     params = urllib.urlencode({'field1': temperatura, 'field2': umidade, 'key': write_key })
27     headers = {"Content-type": "application/x-www-form-urlencoded", "Accept": "text/plain"}
28     conn = urllib.HTTPConnection("api.thingspeak.com:80")
29     #Envia os dados do sensor para o ThingSpeak
30     conn.request("POST", "/update", params, headers)
31     response = conn.getresponse()
32     print response.status, response.reason
33     data = response.read()
34     conn.close()
35     time.sleep(10)

```

Fonte: Autoria Própria (2022)

Após a execução do algoritmo, tem-se como resposta do servidor HTTP do *ThingSpeak* o código 200, que representa que a requisição foi enviada e não teve nenhum erro como resposta, como mostrado na Figura 15.

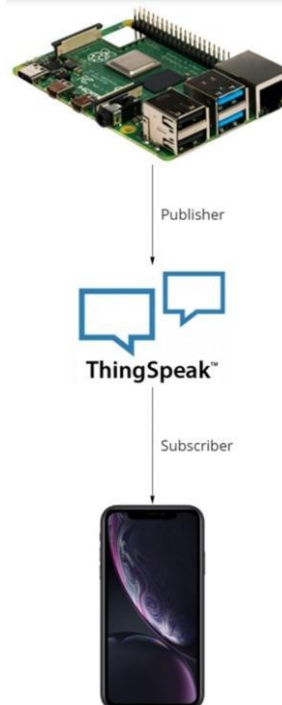
Figura 15 – Execução do Código HTTP

```
pi@raspberrypi: ~/Desktop/HTTP
Arquivo Editar Abas Ajuda
pi@raspberrypi:~/Desktop/HTTP $ python http.py
Lendo a temperatura...
Temperatura = 18.1 Umidade = 66.7%
200 OK
Temperatura = 18.1 Umidade = 67.2%
200 OK
```

Fonte: Aatoria Própria (2022)

3.3 Protótipo IoT com protocolo MQTT

Para o protocolo MQTT a topologia ficou definida da seguinte forma, o Raspberry coleta as informações e publica no servidor através do Broker MQTT do *ThingSpeak*. O celular funciona como cliente recebendo as informações, então a cada nova publicação, o broker envia os dados, como ilustra a Figura 16.

Figura 16 – Topologia MQTT

Fonte: Aatoria Própria (2022)

Assim como foi o desenvolvimento do HTTP, foi criado outro canal, de nome Monitor de Temperatura e Umidade – MQTT como mostrado na Figura 17, para separação dos dados.

Figura 17 – Canal *Thingspeak* com protocolo MQTT



Fonte: Autoria Própria (2022)

Para o servidor receber as informações pelo protocolo MQTT foi criado um meio de nome MQTT autorizando publicar no canal, como mostrado na Figura 18. Com isso tem-se todos os dados necessários para enviar o resultado das medições.

Figura 18 – Permissão para comunicação com o canal



Fonte: Autoria Própria

O código foi mantido a mesma estrutura, mudando apenas a conexão com o servidor e algumas bibliotecas utilizadas, como mostrado na Figura 19.

Figura 19 – Código para comunicação MQTT

```

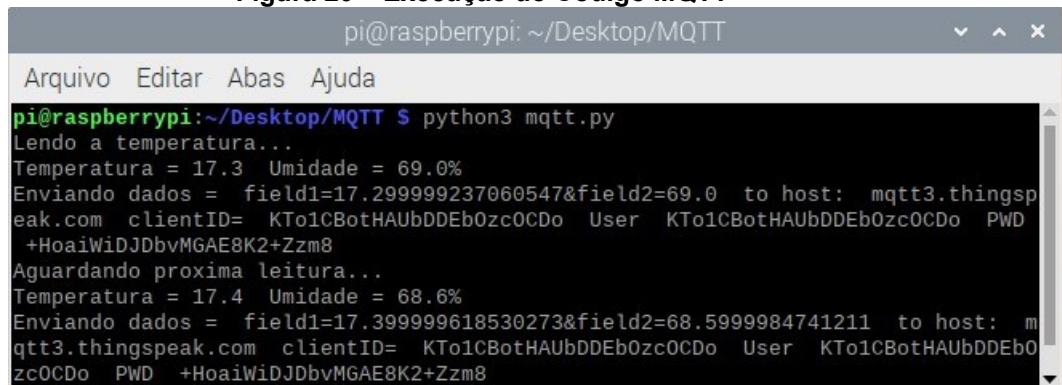
1 # Carrega as bibliotecas
2 import paho.mqtt.publish as publish
3 import psutil
4 import Adafruit_DHT
5 import RPi.GPIO as GPIO
6 import time
7
8 #dados do canal do Thingspeak
9 channel_id = 1529686
10 write_key = 'EZF9BC32KFLCF70A'
11 read_key = 'VHT6QQPJG100I2P3'
12
13 # Define o tipo de sensor = Adafruit_DHT.DHT22
14 sensor = Adafruit_DHT.DHT22
15 GPIO.setmode(GPIO.BOARD)
16
17 # Define a GPIO conectada ao pino de dados do sensor (pino 16)
18 gpio_sensor = 4
19
20 # credenciais MQTT
21 mqtt_host = "mqtt3.thingspeak.com"
22 mqtt_client_ID = "KTo1CBoTHAUb0DEb0zc0CDo"
23 mqtt_username = "KTo1CBoTHAUb0DEb0zc0CDo"
24 mqtt_password = "+H0aiW1Dj0bvMGAERK2+Zzm8"
25 t_port = 1883
26
27 topic = "channels/" + str(channel_id) + "/publish"
28
29 print ("Lendo a temperatura...");
30 while(1):
31     # Efetua a leitura do sensor
32     umidade, temperatura = Adafruit_DHT.read_retry(sensor, gpio_sensor)
33     print ("Temperatura = {0:0.1f} Umidade = {1:0.1f}%".format(temperatura, umidade))
34     dados = "field1="+str(temperatura)+"field2="+str(umidade)
35     try:
36         pprint ("Enviando dados = ", data," to host:", mqtt_host, " clientID= ", mqtt_client_ID, " User ", mqtt_username, " PWD ", mqtt_password)
37         publish.single(topic, dados, hostname=mqtt_host, port=t_port, client_id=mqtt_client_ID, auth={'username':mqtt_username,'password':mqtt_password})
38     except Exception as e:
39         print (e)
40     time.sleep(10)

```

Fonte: Autoria Própria (2022)

Com o algoritmo executado, a Figura 20 mostra os dados e as informações de conexão com o servidor MQTT do *ThingSpeak*.

Figura 20 – Execução do Código MQTT



```

pi@raspberrypi: ~/Desktop/MQTT
Arquivo  Editar  Abas  Ajuda
pi@raspberrypi:~/Desktop/MQTT $ python3 mqtt.py
Lendo a temperatura...
Temperatura = 17.3  Umidade = 69.0%
Enviando dados = field1=17.299999237060547&field2=69.0 to host: mqtt3.thingsp
eak.com clientID= KTo1CBotHAUbDDEb0zc0CDo User KTo1CBotHAUbDDEb0zc0CDo PWD
+HoiWiDJDbvMGAE8K2+Zzm8
Aguardando proxima leitura...
Temperatura = 17.4  Umidade = 68.6%
Enviando dados = field1=17.3999999618530273&field2=68.5999984741211 to host: m
qtt3.thingspeak.com clientID= KTo1CBotHAUbDDEb0zc0CDo User KTo1CBotHAUbDDEb0
zc0CDo PWD +HoiWiDJDbvMGAE8K2+Zzm8
  
```

Fonte: Autoria Própria (2022)

3.4 Visualização dos dados

O *ThingView* foi o aplicativo utilizado para visualização dos dados, devido a sua facilidade de uso, informando o número do canal consegue-se acesso aos canais criados para os testes. Para ter acesso a informações, foi utilizado um celular como cliente (HTTP) e *subscriber* (MQTT).

As Figuras 21 e 22 mostram como ficaram os resultados das capturas utilizando os dois protocolos.

Figura 21 – Cliente HTTP

Fonte: Autoria Própria (2022)

Figura 22 – Cliente MQTT

Fonte: Autoria Própria (2022)

3.5 Análise dos dados obtidos

Os testes foram realizados utilizando a mesma estrutura de código Python como linguagem de programação e o algoritmo sendo rodado em ciclos de 5 minutos. Foi usado o analisador de protocolos *Wireshark* para a captura dos pacotes. Os dados foram filtrados, exportados no formato CSV e analisados no Microsoft Excel.

Durante o tempo de execução foram coletados os pacotes de POST e de resposta do HTTP. Já para o MQTT foram coletados os pacotes de pedido de

autorização ao *broker*, a resposta da autorização, o conteúdo da publicação e a desconexão do broker. Os dados foram coletados apenas na publicação.

Foram realizadas medições em três cenários diferentes, apresentados nas subseções a seguir.

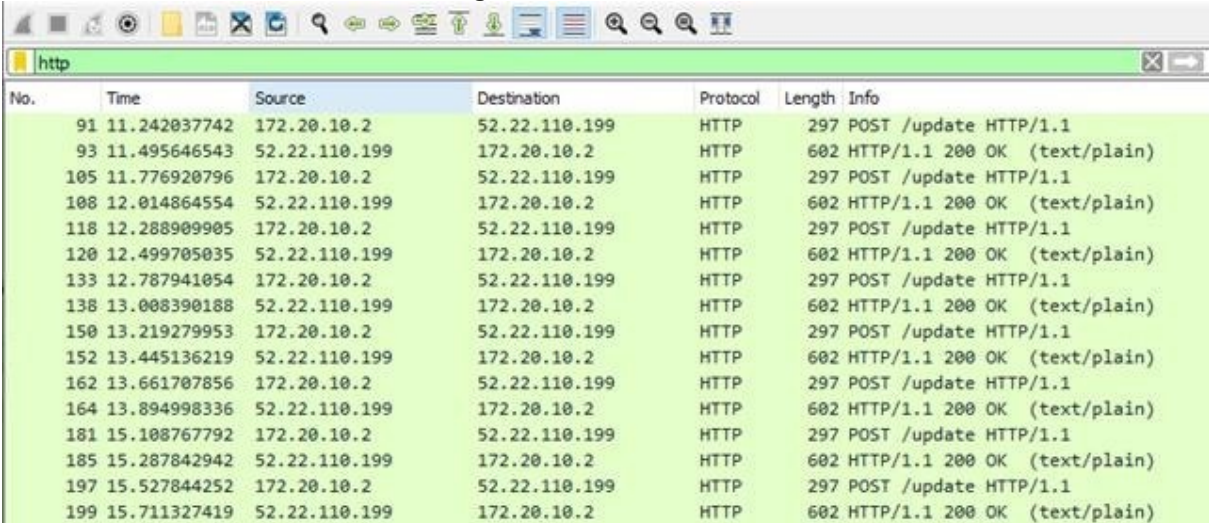
3.5.1 Cenário 1 – Comparação entre HTTP e MQTT com tempo de transmissão de 10 segundos

O objetivo deste cenário é comparar os tamanhos dos pacotes do HTTP e MQTT e a largura de banda necessária para a transmissão dos mesmos.

Neste cenário, o algoritmo foi rodado a cada 10 segundos em um tempo total de 5 minutos. Para a conexão com a internet, foi utilizado a internet fibra.

Cada pacote HTTP pode ser visto separadamente, como mostra a Figura 21. Os pacotes POST contém os dados sendo enviados ao servidor e como resposta o servidor envia o código do estado de sucesso.

Figura 21 - Pacotes HTTP



No.	Time	Source	Destination	Protocol	Length	Info
91	11.242037742	172.20.10.2	52.22.110.199	HTTP	297	POST /update HTTP/1.1
93	11.495646543	52.22.110.199	172.20.10.2	HTTP	602	HTTP/1.1 200 OK (text/plain)
105	11.776920796	172.20.10.2	52.22.110.199	HTTP	297	POST /update HTTP/1.1
108	12.014864554	52.22.110.199	172.20.10.2	HTTP	602	HTTP/1.1 200 OK (text/plain)
118	12.288909905	172.20.10.2	52.22.110.199	HTTP	297	POST /update HTTP/1.1
120	12.499705035	52.22.110.199	172.20.10.2	HTTP	602	HTTP/1.1 200 OK (text/plain)
133	12.787941054	172.20.10.2	52.22.110.199	HTTP	297	POST /update HTTP/1.1
138	13.008390188	52.22.110.199	172.20.10.2	HTTP	602	HTTP/1.1 200 OK (text/plain)
150	13.219279953	172.20.10.2	52.22.110.199	HTTP	297	POST /update HTTP/1.1
152	13.445136219	52.22.110.199	172.20.10.2	HTTP	602	HTTP/1.1 200 OK (text/plain)
162	13.661707856	172.20.10.2	52.22.110.199	HTTP	297	POST /update HTTP/1.1
164	13.894998336	52.22.110.199	172.20.10.2	HTTP	602	HTTP/1.1 200 OK (text/plain)
181	15.108767792	172.20.10.2	52.22.110.199	HTTP	297	POST /update HTTP/1.1
185	15.287842942	52.22.110.199	172.20.10.2	HTTP	602	HTTP/1.1 200 OK (text/plain)
197	15.527844252	172.20.10.2	52.22.110.199	HTTP	297	POST /update HTTP/1.1
199	15.711327419	52.22.110.199	172.20.10.2	HTTP	602	HTTP/1.1 200 OK (text/plain)

Fonte: Autoria Própria (2022)

Já com o protocolo MQTT, pode ser vista a requisição de conexão ao servidor, a resposta do servidor autorizando conexão, a publicação e a desconexão do servidor, como mostrado na Figura 22.

Figura 22 - Pacotes MQTT

No.	Time	Source	Destination	Protocol	Length	Info
2	0.005121876	54.81.146.55	192.168.0.32	MQTT	70	Connect Ack
4	0.007297657	192.168.0.32	54.81.146.55	MQTT	130	Publish Message [channels/1529686/publish]
5	0.008317345	192.168.0.32	54.81.146.55	MQTT	68	Disconnect Req
16	0.288650450	192.168.0.32	54.81.146.55	MQTT	154	Connect Command
18	0.450449636	54.81.146.55	192.168.0.32	MQTT	70	Connect Ack
20	0.452570001	192.168.0.32	54.81.146.55	MQTT	130	Publish Message [channels/1529686/publish]
21	0.453483751	192.168.0.32	54.81.146.55	MQTT	68	Disconnect Req
32	0.745473264	192.168.0.32	18.207.44.162	MQTT	154	Connect Command
37	3.326603146	18.207.44.162	192.168.0.32	MQTT	70	Connect Ack
39	3.328319136	192.168.0.32	18.207.44.162	MQTT	130	Publish Message [channels/1529686/publish]
40	3.328988303	192.168.0.32	18.207.44.162	MQTT	68	Disconnect Req
51	3.608637294	192.168.0.32	18.207.44.162	MQTT	154	Connect Command
53	6.598778268	18.207.44.162	192.168.0.32	MQTT	70	Connect Ack
55	6.600018893	192.168.0.32	18.207.44.162	MQTT	130	Publish Message [channels/1529686/publish]
56	6.600558425	192.168.0.32	18.207.44.162	MQTT	68	Disconnect Req
67	6.875505540	192.168.0.32	54.81.146.55	MQTT	154	Connect Command

Fonte: Autoria Própria (2022)

Nas Tabelas 1 e 2, tem-se a média do tamanho, em bytes, de cada pacote dos dois protocolos.

Tabela 1 - Média de tamanho dos pacotes HTTP

Pacotes HTTP	Tamanho pacote (bytes)
POST /update HTTP/1.1	297
HTTP/1.1 200 OK (text/plain)	602

Fonte: Autoria Própria (2022)

Tabela 2 - Média de tamanho dos pacotes MQTT

Pacotes MQTT	Tamanho pacote (bytes)
Connect Command	154
Connect Ack	70
Publish Message	130
Disconnect Req	68

Fonte: Autoria Própria (2022)

Como é possível observar, por mais que o protocolo MQTT tenha mais pacotes por requisição ao broker do que o HTTP ao servidor, o tamanho dos pacotes é consideravelmente menor.

Para determinar a largura de banda necessária para transmitir os pacotes de cada protocolo, é somado o tamanho de todos os pacotes enviados e recebidos e divididos pelo tempo em que o algoritmo ficou enviando os dados, que neste teste é de 5 minutos. Ambos os protocolos enviaram 19 dados totais, como o HTTP enviou

um e recebeu um foram 38 pacotes totais, e o MQTT teve 4 pacotes a cada ciclo, foram totalizados 76 pacotes. O resultado é mostrado na Tabela 3.

Pode-se verificar que o MQTT necessita de uma largura de banda de 219 bps e o HTTP necessita de uma largura de banda de 695 bps para transmitir um dado de temperatura e umidade a cada 10s.

Tabela 3 - Comparação da vazão entre HTTP e MQTT

Protocolo	Quantidade de pacotes	Soma do tamanho dos pacotes (bytes)	Largura de Banda (bps)
HTTP	38	15238	695
MQTT	76	8349	219

Fonte: Autoria Própria (2022)

3.5.2 Cenário 2 – Comparação entre HTTP com MQTT com tempos de transmissão de dados diferentes

O objetivo deste cenário é mostrar a necessidade de largura de banda com os dois protocolos à medida que o intervalo de transmissão diminui.

O algoritmo foi rodado em diferentes intervalos de tempo: 10, 1s, 0.1s e 0.01s em um tempo total de 5 minutos. Para a conexão com a internet, foi utilizado a internet fibra.

Tabela 4 - Comparação da vazão de protocolos HTTP com tempos de transmissão diferentes

Tempo de envio de pacotes com HTTP (s)	Largura de Banda (bps)
10	695,62
1	5122,08
0,1	15405,06
0,01	18804,88

Fonte: Autoria Própria (2022)

Tabela 5 - Comparação da vazão de protocolos MQTT com tempos de transmissão diferentes

Tempo de envio de pacotes com MQTT (s)	Largura de Banda (bps)
10	219,33
1	1013,84
0,1	1624,58
0,01	2306,18

Fonte: Autoria Própria (2022)

Percebe-se que a largura de banda necessária para a transmissão dos dados aumenta à medida que o tempo de transmissão diminui. E que o protocolo MQTT necessita de uma largura de banda menor do que o HTTP. Para o intervalo de tempo de 0,01s, o MQTT precisa de 2,3 kbps e o HTTP precisa de 18,8 kbps. Isso é quase 8 vezes menor. É importante este tipo de análise porque sabendo a largura de banda necessária para um projeto de IoT, pode-se especificar o meio de transmissão adequado, que é o objetivo do cenário 3, onde são analisados 3 meios de transmissão. A largura de banda total de um projeto de IoT depende do tempo de transmissão do dado e da quantidade de sensores. Se este projeto tiver 100 sensores e a transmissão for a cada 0,01s, a largura de banda necessária será de 230 kbps para o MQTT e 1,88 Mbps para o HTTP.

3.5.3 Cenário 3 – Comparação dos protocolos utilizando internet fibra e internet móvel

No último cenário, o algoritmo foi rodado em diferentes intervalos de tempo: 10s, 1s, 0.1s e 0.01s em um tempo total de 5 minutos. Para a conexão com a internet foi utilizado a internet móvel do celular e comparado com os dados obtidos anteriormente pela conexão com a internet fibra.

A tabela 6 apresenta a diferença de vazão de internet entre os diferentes tipos de conexão à internet.

Tabela 6 - Comparação da vazão entre Internet Fibra, 4G e 3G

Conexão	Vazão download (Mbps)	Vazão upload (Mbps)
Fibra	242	27
4G	13	9,8
3G	7,6	4,2

Fonte: Autoria Própria (2022)

A latência foi calculada dividindo-se o tamanho das mensagens de cada protocolo mostrados nas Tabelas 1 e 2 pela vazão de cada conexão à internet mostrada na Tabela 6.

Tabela 7 – Média de latência das mensagens HTTP na rede Fibra

Mensagem HTTP	Latência (us)
POST	10
HTTP OK	178

Fonte: Autoria Própria (2022)

Tabela 8 – Média de latência das mensagens MQTT na rede Fibra

Mensagem MQTT	Latência (us)
<i>Connect Command</i>	45
<i>Connect Ack</i>	2,3
<i>Publish Message</i>	38
<i>Disconnect Req</i>	20

Fonte: Autoria Própria (2022)

Para comparar a latência dos dois protocolos, foi somada a latência de cada mensagem. Desse modo, a latência com o protocolo HTTP é de 188us e do MQTT é de 105,3us com internet fibra.

Tabela 9 – Média de latência das mensagens HTTP na rede 4G

Mensagem HTTP	Latência (us)
POST	182
HTTP OK	491

Fonte: Autoria Própria (2022)

Tabela 10 – Média de latência das mensagens MQTT na rede 4G

Mensagem MQTT	Latência (us)
<i>Connect Command</i>	125
<i>Connect Ack</i>	43
<i>Publish Message</i>	106
<i>Disconnect Req</i>	55

Fonte: Autoria Própria (2022)

Com internet 4G, a latência com o protocolo HTTP é de 673 us e com o protocolo MQTT é de 329 us.

Tabela 11 – Média de latência das mensagens HTTP na rede 3G

Mensagem HTTP	Latência (us)
POST	354
HTTP OK	1146

Fonte: Aatoria Própria (2022)

Tabela 12 – Média de latência das mensagens MQTT na rede 3G

Mensagem MQTT	Latência (us)
<i>Connect Command</i>	293
<i>Connect Ack</i>	83
<i>Publish Message</i>	247
<i>Disconnect Req</i>	129

Fonte: Aatoria Própria (2022)

Com internet 4G, a latência com o protocolo HTTP é de 1500 us e com o protocolo MQTT é de 752 us.

Analisando os dados coletados nas tabelas acima, percebe-se que a latência das mensagens aumenta conforme a qualidade da conexão com a internet diminui. E que o protocolo MQTT possui uma latência menor do que o HTTP.

4 CONCLUSÃO

Para ter um projeto de IoT completo e eficiente é preciso analisar as condições, o ambiente e as necessidades de cada projeto. Este trabalho é voltado a uma pequena parte deste projeto, mais especificamente uma análise entre os dois protocolos mais utilizados para este objetivo, o HTTP e MQTT.

Sendo assim, primeiramente, no capítulo 2, foi feita uma abordagem sobre a teoria do que é Internet das Coisas e como é definida a sua arquitetura, com uma análise maior nos protocolos de comunicação e uma comparação, além de uma breve explicação do controlador utilizado neste trabalho, encontradas e citadas na literatura. No capítulo 3, os objetivos específicos foram representados, através do desenvolvimento do protótipo IoT com os protocolos HTTP e MQTT, a medição dos pacotes enviados e recebidos utilizando cada protocolo para o propósito de calcular a vazão e a latência com três diferentes conexões a internet: fibra, 4G e 3G. Também neste capítulo foi tratado dos resultados, para cada protocolo utilizado, assim como uma comparação entre eles.

Pode-se concluir que a largura de banda necessária para um projeto de IoT com MQTT é menor comparado com o HTTP. A latência para a transmissão das mensagens usando o MQTT também é menor, se mostrando mais adequado para aplicações que necessitem uma baixa latência. E por último, pode-se concluir que a rede de comunicação influencia diretamente na latência, sendo que uma internet mais rápida é mais adequada para aplicações de tempo real.

REFERÊNCIAS

- ALECRIM, Emerson. **O que é Internet das Coisas (IoT)?**. Disponível em: <https://www.infowester.com/iot.php>. Acesso em: 10 set. 2022.
- AVELAR, Edson Adriano M.; AVELAR, Lorena M.; DIAS, Kelvin Lopes; SILVA, Diego dos Passos. **Arquitetura de Comunicação para Cidades Inteligentes: Uma proposta heterogênea, extensível e de baixo custo**. Disponível em: http://projeto.unisinos.br/simtur/papers/Kelvin_ARTIGO_2012.pdf. Acesso em: 01 set. 2022.
- ASHTON, Kevin. **That ‘Internet of Things’ Thing**. *RFID Journal*. 2009. Disponível em: <https://www.rfidjournal.com/articles/view?4986>. Acesso em: 02 set. 2022.
- BARRIOS QUINTANILLA, Alejandro; SILVA FILHO, Deijaval Pereira da. **Ataques de negação de serviço na camada de aplicação: estudo de ataques lentos ao protocolo HTTP**. 2015. 56 f., il. Monografia (Bacharelado em Engenharia de Redes de Comunicação)- Universidade de Brasília, Brasília, 2015.
- CASTRO, Arthur Victor. **Fundamentos de internet das coisas - IOT**. 2019. Trabalho de Conclusão de Curso na Faculdade IDAAM, Manaus, 2019.
- CARRION, P.; QUARESMA, M. Internet da Coisas (IoT): Definições e aplicabilidade aos usuários finais. *Human Factors in Design*, Florianópolis, v. 8, n. 15, p. 049-066, 2019. DOI: 10.5965/2316796308152019049. Disponível em: <https://www.revistas.udesc.br/index.php/hfd/article/view/2316796308152019049>. Acesso em: 8 jul. 2022.
- CRAGGS, IAN. **MQTT Vs. HTTP for IoT**. Disponível em: <https://www.hivemq.com/blog/mqtt-vs-http-protocols-in-iiot/#:~:text=They%20both%20run%20over%20TCP,respond%20to%20requests%20from%20clients>. Acesso em: 8 jul. 2022.
- Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., “**HTTP Semantics**”, STD 97, RFC 9110, DOI 10.17487/RFC9110, Junho 2022, Disponível em: <https://www.rfc-editor.org/info/rfc9110>. Acesso em: 8 jul. 2022.
- GRÜDTNER, Lukas Derner. **Desenvolvimento de uma aplicação IoT utilizando CoAP e DTLS para telemetria veicular**. 2020. Trabalho de Conclusão de Curso (Ciências da Computação) - Universidade Federal de Santa Catarina, [S. I.], 2020.
- HALFACREE, Gareth; UPTON, Eben. **Raspberry Pi: Guia do Usuário**. 4ª edição. ed. [S. I.]: Alta Books, 2018.

INTERNACIONAL BUSINESS MACHINES CORPORATION; EUROTECH. **MQTT specification version 3.1**. out. 2010. Disponível em:

<public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html> . Acesso em 6 set. 2021.

INTERNATIONAL CONFERENCE ON PHYSICAL INSTRUMENTATION AND

ADVANCED MATERIALS, 853., 012003, Indonésia. IoT real time data acquisition using MQTT protocol [...]. [S. l.: s. n.], 2017. DOI 10.1088/1742-6596/853/1/012003.

Disponível em: <http://iopscience.iop.org/1742-6596/853/1/012003>. Acesso em: 6 ago. 2021.

KUROSE,J.F.;ROSS,K.W.;**Redes de computadores e a internet:uma abordagemTop-down**.6.ed.SãoPaulo:Pearson,2013

LEITE,J.R. Emiliano.; MARTINS,Paulo S.; URSINI, EDSON L., 2019, Campinas - SP. **Segurança e Gerenciamento da Rede IoT** [...]. [S. l.: s. n.], 2019. Disponível em: https://www.researchgate.net/publication/336903352_Seguranca_e_Gerenciamento_da_Rede_IoT. Acesso em: 6 set. 2022.

LUVISOTTO, M – **Ultrahigh-Performance Wireless Control for Critical Applications: Challenges and Directions** – IEEE Transactions on Industrial Informatics, 2016.

MONK , Simon. **Raspberry Pi Cookbook**. 2ª edição. ed. [S. l.]: O'Reilly, 2016.

MQTT. **Tutorials: Sensors, interfaces and bus systems (SENIN, BUSSY)**. Disponível em: https://www.weigu.lu/tutorials/sensors2bus/06_mqtt/index.html. Acesso em: 6 set. 2022.

OLIVEIRA, Sérgio. **Internet das Coisas com ESP8266, Arduino e Raspberry Pi**. Primeira Edição. ed. [S. l.]: Novatec Editora Ltda, 2017. 257 p. ISBN 978-85-7522-582-0..

REVELL, S. **Internet of Things (IoT) and Machine to Machine Communications (M2M) Challenges and Opportunities**. Final Paper, London, UK Google Scholar, 2013.

SILVA JUNIOR, Mauricio Pavan da. **Análise entre protocolos HTTP e MQTT em projetos IOT**. 2021. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) - Universidade Tecnológica Federal do Paraná, Ponta Grossa, 2021.

SOUZA, Jaime F. **Avaliação do Protocolo HTTP 2.0**. Orientador: Prof. Me. Stenio Longo Araújo. 2015. 58 f. Monografia (Graduação em Ciência da Computação) - Universidade Estadual do Sudoeste da Bahia, Vitória da Conquista-BA, 2015..

TELECO.**Velocidade 4G**. Disponível em:
https://www.teleco.com.br/4g_velocidade.asp. Acesso em: 10 set. 2022

TISOTT, C. G. **Uma revisão sistemática da literatura a respeito do uso da tecnologia de comunicação 5G na infraestrutura de comunicação das aplicações de resposta à demanda e de infraestrutura de medição avançada em Smart Grids**. repositorio.ifsc.edu.br. 2021. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Elétrica) – Instituto Federal de Santa Catarina, Itajai, 2021.

YASSEIN, Muneer Bani et al. **Internet of Things: Survey and open issues of MQTT protocol**. In: IEEE. 2017 International Conference on Engineering & MIS (ICEMIS). [S.l.: s.n.], 2017. p. 1–6.