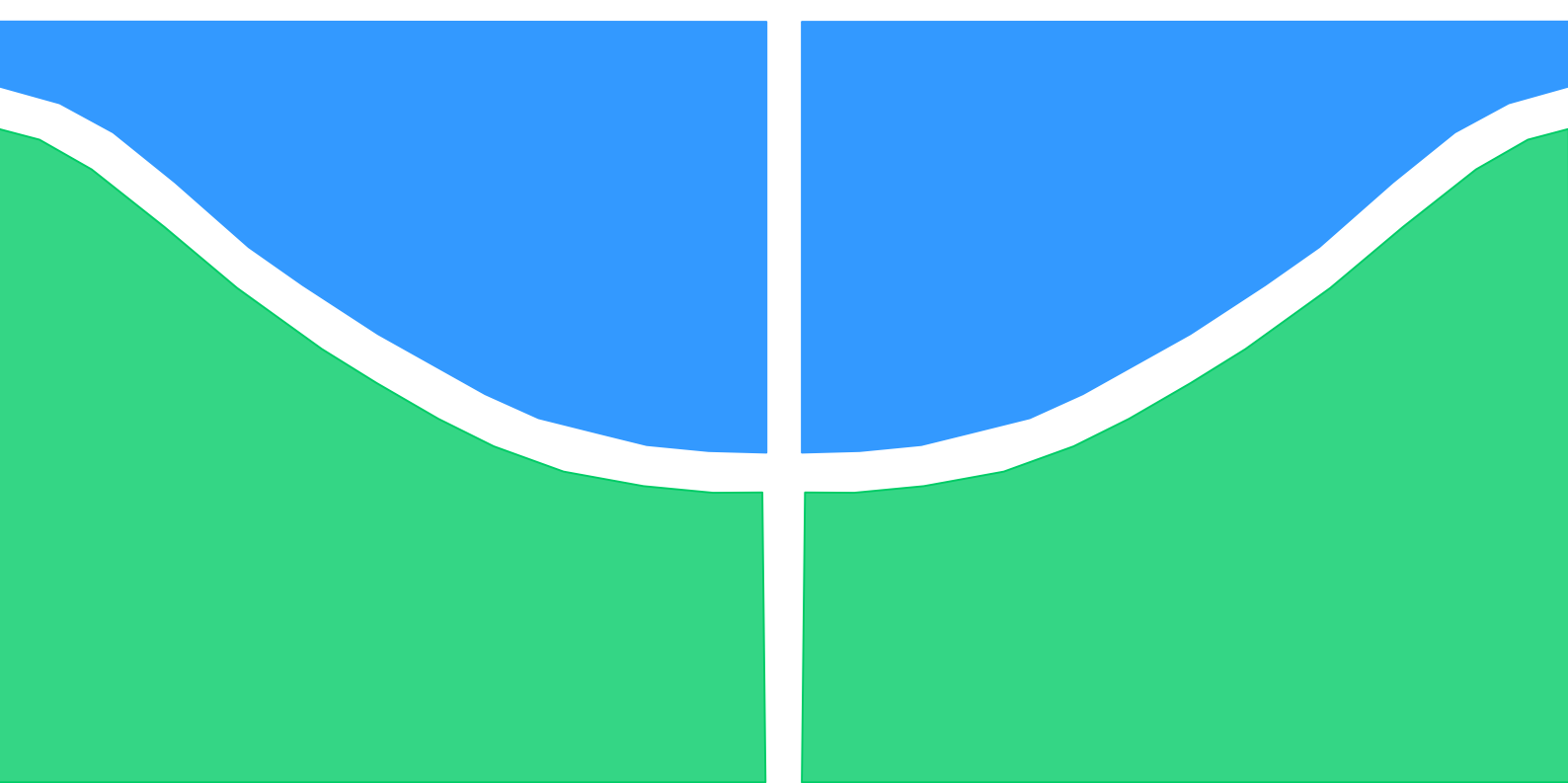




TRABALHO DE GRADUAÇÃO

Proposta de controle de nível de acesso para o protocolo MQTT na comunicação de dispositivos de uma rede baseada em Internet das Coisas.

**Matheus Fontinele dos Santos Bessa
Renan Bittencourt Coelho Scorza Gonçalves**

Brasília, Dezembro de 2017

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

TRABALHO DE GRADUAÇÃO

Proposta de controle de nível de acesso para o protocolo MQTT na comunicação de dispositivos de uma rede baseada na Internet das Coisas.

Matheus Fontinele dos Santos Bessa
Renan Bittencourt Coelho Scorza Gonçalves

Trabalho de conclusão de curso submetido ao departamento de Engenharia Elétrica da Faculdade de Tecnologia da Universidade de Brasília como parte dos requisitos necessários para a obtenção do grau de Engenheiro de Redes de Comunicação.

Banca Examinadora

Prof. Dr. Edgard Costa Oliveira, UnB/Faculdade UnB
Gama/Engenharia de Software

Prof. Beatriz Campos Santana de Araujo UNB/Faculdade de
Tecnologia/Departamento Engenharia Elétrica

Eng. MSc. Rodrygo Torres Cordova, RNP/ Brazilian NREN
(National Research and Education Network)

À todos que torcem pelo nosso sucesso.

AGRADECIMENTOS

RESUMO

PROPOSTA DE CONTROLE DE NÍVEL DE ACESSO PARA O PROTOCOLO MQTT NA COMUNICAÇÃO DE DISPOSITIVOS DE UMA REDE BASEADA NA INTERNET DAS COISAS

Autores: Matheus Fontinele dos Santos Bessa, Renan Bittencourt Coelho Scorza Gonçalves

Orientador: Prof. Doutor Edgard Costa Oliveira, FGA/UnB

**Projeto Final de Graduação
Brasília, XX de Dezembro de 2017**

Com a evolução da internet e grandes avanços nas áreas de comunicação e microeletrônica, o conceito de computação ubíqua vem chegando cada vez mais próximo da realidade. Está cada vez mais visível, objetos que, costumeiramente se utilizam estão ganhando a capacidade de se conectarem a internet e modificando a forma de interação entre homem e objeto. Relógios possuem funções que vão além de marcar as horas, uma simples pulseira, antes utilizada como acessório de moda, torna-se seu enfermeiro pessoal. A Internet das Coisas oferece enormes oportunidades para revolucionar o modo de realização das nossas atividades em diversas áreas, como assistência médica, logística ou automação. O fato é que as aplicações desenvolvidas manipulam dados sensíveis, e qualquer intromissão pode gerar grandes perdas financeiras ou até mesmo atentar contra a integridade física dos usuários.

Dessa forma, um dos grandes desafios para quem trabalha na construção de serviços nessa área é desenvolver aplicações suficientemente seguras para evitar, vazamento de informações, ou comprometimento dos serviços. Já temos uma série de relatos sobre as violações de segurança em soluções para Internet das Coisas, por isso neste trabalho iremos abordar o funcionamento do protocolo MQTT, que vem sendo adotado em inúmeras soluções e pode tornar-se o padrão vigente para a comunicação entre dispositivos e deles com serviços, e modificá-lo para que seja capaz de fornecer a funcionalidade de controle de acesso, permitindo que o gerente da rede limite as permissões de cada usuário, baseado no princípio publish/subscribe do MQTT, afim de obter um grau maior de segurança e contribuir para um ambiente mais estável e menos suscetível a ataques para a Internet das Coisas.

ABSTRACT

PROPOSED CONTROL OF LEVEL OF ACCESS FOR THE MQTT PROTOCOL ON THE COMMUNICATION OF DEVICES FROM A NETWORK BASED ON THE INTERNET OF THINGS

Authors: Matheus Fontinele dos Santos Bessa, Renan Bittencourt Coelho Scorza Gonçalves

Supervisor: Prof. Doutor Edgard Costa Oliveira, FGA/UnB

Final Graduation Project

With the evolution of the internet and great improvement in the communication and microelectronics area, the concept of Ubiquitous Computing is getting closer to reality. It's increasingly visible that the objects which we use customarily are gaining capability of being connected to the internet and modifying the relationship between man and object. Watches are exhibiting functions that go beyond telling the time or being a simple bracelet that was once seen as a fashion accessory, they are now personal nurses. The Internet of Things offers tremendous opportunities to revolutionize the way of accomplishing our tasks in many different areas, like medical assistance, logistics or automation. The fact is that the applications developed manipulate sensitive data, and any intrusion may generate great financial losses or even attack the physical integrity of its users. As a result, one of the biggest challenges for those who work in this area's construction services is to develop applications which are sufficiently safe in order to avoid the leakage of information or damages to the services. We already have a series of reports on safety violations of the Internet of Things, therefore, in this essay, we will debate the functioning of the MQTT protocol, which is being chosen in several solutions and may become the standard in force for the communication between gadgets and between them and the services, and modify it to reach the capability of providing the functionality of access control, allowing the network manager to limit the permissions of each user, based on the MQTT's publish/subscribe principle, in order to obtain greater safety and contribute to the achievement of a more stable and less susceptible to attacks environment for the Internet of Things.

SUMÁRIO

1. INTRODUÇÃO	1
1.1. Motivação.....	2
1.2. Objetivos	2
1.3. Metodologia de pesquisa.....	3
2. ESTADO DA ARTE E REVISÃO BIBLIOGRÁFICA	4
2.1. Internet das Coisas.....	4
2.1.1. Elementos da Internet das Coisas.....	7
2.1.2. Modelo básico de Arquitetura para Internet das Coisas em camadas.....	10
2.2. Tecnologias Relacionadas	12
2.2.1. Tecnologias de Rede	12
2.2.2. Protocolos da Camada de Aplicação.....	16
2.2.3. Segurança	18
2.2.4. Tecnologias de prototipagem de dispositivos	24
3. Message Queue Telemetry Transport (MQTT)	26
3.1. Formato da mensagem MQTT	27
3.1.1. Cabeçalho Fixo	27
3.1.2. Tipos de mensagens MQTT	28
3.2. Fluxos MQTT e Níveis de Serviço (QoS)	31
4. PROPOSTA DE ALTERAÇÃO DO PROTOCOLO MQTT.....	34
4.1. Ambiente de simulação	34
4.2. Controle de Nível de Acesso.....	36
4.2.1. Estabelecendo os limites dos tópicos	37
4.2.2. Credenciais de acesso.....	37
4.1. Função de Registro.....	37
4.1.1. Formato mensagem CONNECT	38
4.2. Broker.....	39
4.2.1. Recebendo solicitações de conexão	39
4.2.2. Análise do Contexto.....	42
5. Conclusão.....	44
6. REFERÊNCIAS BIBLIOGRÁFICAS.....	45

LISTA DE FIGURAS

Figura 1 - Relação Computação Pervasiva e Computação Móvel(ARAÚJO,2003)	5
Figura 2 – Elementos de uma rede IoT(Fonte: SANTOS,2016).....	9
Figura 3 - Modelo básico de 3 camadas para Internet das Coisas(SANTOS, 2016)	11
Figura 4 - Representação funcionamento básico MQTT, princípio publish/subscribe .(HIVEMQ.COM,2016).....	26
Figura 5 - Fluxo de mensagens MQTT com QoS nível 0	31
Figura 6 - Fluxo de mensagens MQTT com QoS nível 1	31
Figura 7 - Fluxo de mensagens MQTT com QoS nível 2.....	32
Figura 8 - Topologia utilizada na simulação.....	34
Figura 9 - Captura wireshark (conexão entre maquina e broker)	35
Figura 10 - Captura wireshark(conexão entre aplicação e broker)	35
Figura 11 - Captura wireshark (broker enviando mensagem para aplicação).....	35
Figura 12 - Captura wireshark (conexão entre maquina e broker com QoS 1)	36
Figura 13 - Captura wireshark (conexão entre maquina e broker com QoS 2)	36
Figura 14 - Fluxograma da função registro.....	41

LISTA DE TABELAS

Tabela 1 - Vulnerabilidades da Internet das Coisas	19
Tabela 2 - Cabeçalho fixo mensagem MQTT.....	27
Tabela 3 - Tipos de mensagens MQTT	28
Tabela 4 - Respostas para mensagem CONNECT.....	29

1. INTRODUÇÃO

É evidente que a tendência para o futuro é que cada objeto, independente da sua natureza, esteja conectado à internet. Aparelhos com capacidade de interagir com o ambiente, com os seres humanos e com outros objetos. Essa ideia segue um conceito antigo apresentado por Marc Weiser, conhecido como Computação Ubíqua que hoje é representada pelo movimento que está sendo denominado de Internet das Coisas, tradução de *Internet of Things* – *IoT*. Idealizada há tempos, mas só agora vem se tornando realidade devido à evolução de diversas áreas que em conjunto fornecem os insumos para criar uma rede de objetos inteligentes. Avanços na eletrônica, na produção de sistemas embarcados e a evolução das tecnologias de comunicação contribuíram nesse processo. O mais incrível desse paradigma é a quantidade de aplicações possíveis, para serem postas em prática.

É impressionante a mudança que a IoT pode realizar no cotidiano da sociedade, nas atividades mais simples, como ligar uma lâmpada, por exemplo, ou em operações mais complexas, como monitorar um idoso e controlar a dosagem das medicações necessárias ao longo do dia. Todavia, o fato dessas tecnologias estarem bem mais próximas dos seres humanos e passarem a ter o controle de diversas atividades cruciais, ou não, para vida humana, trazem alguns desafios que, necessariamente precisam ser vencidos para estabelecer a Internet das Coisas como o padrão a ser seguido para o futuro. Em alguns ambientes poderemos ter objetos afastados geograficamente, alimentados por baterias. O que nos impõe a regra de sempre implementar soluções, pensando no consumo consciente dos recursos que são limitados, encontrar restrições de energia, possivelmente, links de comunicação também serão problemas, devido a altas taxas de latência e incrementos de erro na comunicação. Entraremos, também, em cenários onde máquinas administram serviços críticos, como um sistema de distribuição de energia elétrica, ou o controle dos estoques de uma multinacional.

Imagine agora, se alguém resolver interferir no funcionamento desses sistemas, isso poderia gerar grandes problemas, de perdas financeiras para empresas até o impacto direto na qualidade de vida de habitantes de uma cidade inteira. Percebemos, assim, que o quesito segurança, agora mais do que antes, tem elevado grau de importância, sendo este o objeto de estudo do nosso trabalho.

Esta pesquisa visa propor um modelo de segurança para o protocolo de comunicação MQTT, acrescentado o controle de níveis de acesso ao funcionamento do protocolo. Com o intuito de incrementar o grau de segurança que atualmente é oferecido pelo MQTT.

1.1. Motivação

Com o advento da Internet das Coisas, a tendência é que dispositivos providos de capacidade computacional tornem-se responsáveis pela execução de atividades do nosso cotidiano, complexas ou não, e com diversos níveis de criticidade. Independente da natureza da atividade o fato é que junto com a comodidade oferecida pelos avanços tecnológicos, temos o risco de que alguém mal-intencionado possa interferir no andamento das tarefas. Como foi dito, dependendo do grau de importância dessa atividade, as consequências podem tomar proporções desastrosas por não serem simples ferramentas para atividades como editar textos ou ouvir música. As ações tomadas por máquinas são sentidas agora no mundo real colocando até mesmo a integridade física dos seres humanos em risco. Desta forma, o quesito segurança é de grande relevância para determinar o futuro da IoT. Sendo assim, se optou por contribuir nessa área que detêm grandes desafios e que será crucial para determinar o sucesso da Internet das Coisas.

1.2. Objetivos

Geral:

Apresentar uma proposta de alteração do protocolo Message Queuing Telemetry Transport –MQTT capaz de controlar o acesso a informações e o grau de liberdade de dispositivos em uma rede IoT.

Específicos

Caracterizar um ambiente de IoT, mostrando às diversas partes que trabalhando em conjunto possibilitam o conceito de que tudo pode estar conectado à internet possa virar realidade. Especificação de um ambiente de IOT para proposição de uma solução de segurança.

Apresentar a estrutura do protocolo MQTT, de acordo com suas especificações e suas principais características.

Oferecer uma proposta de alteração para o protocolo, de modo que possa ser realizado o controle de acesso dos dispositivos para produzir ou consumir informações geradas e que servem de base para as ações tomadas pelo sistema, levando em consideração a análise do contexto que está envolta.

1.3. Metodologia de pesquisa

O modo de pesquisa foi dividido em etapas para sistematizar o desenvolvimento do trabalho. Essa metodologia visa organizar o estudo relacionado à área de segurança de uma rede IoT e os principais problemas que envolvem o tema supracitado. Identificar os principais tópicos conversados no meio acadêmico e de que forma eles podem contribuir para pesquisa.

- Identificar e analisar as principais características, ameaças e vulnerabilidades de segurança em ambientes IoT, bem como, as principais recomendações de segurança para o mesmo ambiente, por meio de pesquisas bibliográficas em livros de literatura especializada e artigos acadêmicos, base de dados. Além de realizar uma compilação dos estudos já realizados na área a fim de identificar os avanços já alcançados e montar um panorama do estado da arte da área de segurança da IoT.
- Levantar a atual estrutura do protocolo MQTT, na sua especificação mais atual.
- Formular proposta de alteração para o protocolo MQTT, que possa realizar o controle de níveis de acesso de aplicações e equipamentos que utilizam o protocolo na camada de aplicação para trocar mensagens.

2. ESTADO DA ARTE E REVISÃO BIBLIOGRÁFICA

Neste capítulo foi abordado os principais conceitos que envolvem a Internet das Coisas, e as tecnologias utilizadas na sua formação. Suas principais características e sobre aspectos que envolvem a área de segurança.

2.1. Internet das Coisas

No ano de 1991, Mark Weiser, chefe do Laboratório de Ciências da Computação do Centro de Pesquisa de Palo Alto (PARC) da empresa Xerox, definiu em seu artigo escrito para a revista especializada *Scientific american*, o termo Computação Ubíqua. No artigo *The Computer for the 21st Century*, é compartilhada a ideia de que as tecnologias mais marcantes e duradouras são aquelas que passam despercebidas e estão misturadas ao cotidiano de quem às utilizam [WEISER, 1991]. Para Weiser a computação ubíqua representa o estado onde os mais variados dispositivos possuem poder computacional, auxiliam os seres humanos em suas atividades, mas são invisíveis, dessa forma podemos nos concentrar na atividade que será realizada e não em como utilizar o dispositivo.

A computação ubíqua representa a mudança da forma que nos relacionamos com as máquinas, que deixarão de ser estações de trabalho, notebooks, celulares e passarão a ser ferramentas até para as atividades mais simples da nossa vida cotidiana [DE ARAUJO, 2003]. Para cumprir com seu objetivo o paradigma da ubiquidade é guiado por três princípios:

- **Diversidade:** Carrega o conceito de que cada dispositivo desempenha, unicamente, a função para qual foi desenvolvido.
- **Conectividade:** Além de prestar um único serviço, o dispositivo deve ser capaz de trocar informações com qualquer outro, compartilhar seus estados ou controlar outros dispositivos, independente da sua localização.
- **Descentralização:** Garante que para desempenhar sua função, o dispositivo não depende de nenhuma outra entidade.

Podemos definir também a computação ubíqua como a junção de dois outros conceitos, computação móvel e computação pervasiva. Na computação móvel encontra-se as tecnologias necessárias que possibilitam a locomoção durante a utilização de serviços computacionais, que garanta ao usuário o acesso as funções disponíveis na rede, independente da sua localização ou do momento em que o serviço é requisitado. Este paradigma tem como principal apoio as tecnologias de comunicação sem fio, que elimina a necessidade, anteriormente existente de estabelecer uma conexão física com a

infraestrutura de rede através de cabos, obrigando que o dispositivo mantivesse uma posição fixa no ambiente. Já a computação pervasiva abrange a capacidade de sistemas estarem espalhados e imersos no ambiente, de forma com que o usuário não o perceba, além disso, devem possuir a habilidade de obter informações do meio que se encontram, adaptando-se e tomando decisões, para atender da melhor maneira possível as requisições feitas às aplicações, bem como a capacidade de perceber outros dispositivos e de interagir com eles. Formando um ambiente integrado e inteligente capaz de entender, controlar e configurar o ambiente. Na interseção desses dois mundos aparece a computação ubíqua, que se baseia em um modelo computacional capaz de atuar nos mais variados cenários, tomar decisões, reconhecer outros dispositivos e o ambiente ao seu redor mesmo que se esteja em movimento[DE ARAUJO, 2003]. Isso, atrelado ao conceito apresentado por Weiser de que o computador deve ser inserido no ambiente de forma com que o dispositivo em si não encubra a sua função, podemos perceber a inversão de papéis entre humanos e máquinas, antes o homem mergulhava no ambiente virtual e agora a máquina invade o nosso espaço e interfere no mundo real. Seguindo esses princípios é possível montar o cenário imaginado por Weiser no final dos anos 80, onde existem milhares de funcionalidades, cada uma sendo exercida por um dispositivo específico, servindo aos seres humanos sem que estes percebam a presença das máquinas [FERREIRA, 2014]. Weiser já havia previsto que no futuro estariam disponíveis uma enorme quantidade de serviços, e sua visão dia após dia vêm se concretizando. Uma pesquisa da Cisco prevê que até 2020 chegaremos à quantidade de 50 bilhões de dispositivos conectados à internet [EVANS, 2011].

Nesse contexto, em conjunto com os milhares de novos dispositivos que se conectam a cada dia, surge o que é chamado de Internet das Coisas, mais conhecida pela sigla do termo em inglês *Internet of Things(IoT)*, onde a idéia básica deste conceito é a presença de uma variedade de objetos interconectados, capazes de interagir e trabalhar para atingir objetivos em comum[ATZORI,2010]. O termo foi citado pela primeira vez em 1999, por Kevin Ashton em uma apresentação, para executivos da Proctor & Gamble [ASHTON, 1999], sobre uso da tecnologia

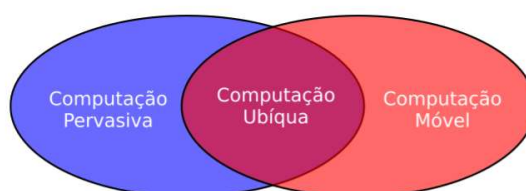


Figura 1 - Relação Computação Pervasiva e Computação Móvel(ARAUJO,2003)

LRFID para o controle da logística e do estoque da multinacional. O conceito de IoT já era vislumbrado por Weiser décadas atrás, entretanto tornou-se possível com o avanço de diversas áreas

como sistemas embarcados, microeletrônica, comunicação e tecnologia de informações [SANTOS,2016]. Permitindo, assim, que objetos antes isolados, desprovidos de capacidade de processamento, agora sejam sensíveis ao contexto e possam interagir com outros objetos através de redes de comunicação, capturar, processar e apresentar informações e tomar decisões, nascendo assim os *Smart Objects*. Gerando uma gama enorme de aplicações e uma nova forma de interação. Além do fluxo já bem conhecido entre humanos e máquinas, surge a comunicação de máquina para máquina e com o ambiente a sua volta. [LACERDA, 2015]. Levando em consideração os números apresentados por Evans, podemos ter uma idéia das proporções que esse novo conceito pode tomar, e nos preparar para o que podemos considerar como uma revolução no mundo da Tecnologia da Informação [SANTOS, 2016].

Como já foi citado anteriormente, o avanço de diversas áreas permitiu a ascensão das ideias que hoje são pregadas pela Internet das Coisas para o centro das pesquisas tecnológicas. Sendo possível destacar os principais tópicos para desenvolver as tecnologias que visam seguir os preceitos da IoT. Sem dúvida, deve-se fazer referências ao enorme salto de desenvolvimento apresentado nas últimas décadas dos sistemas de comunicação, passando das ligações feitas por fio até os atuais e avançados modelos de comunicação sem fio, que trazem mobilidade para rede e tornam muito mais prático a organização do ambiente. Caso contrário, se cada dispositivo previsto por Evans necessitasse de um fio para se conectar à rede, inviabilizaria o projeto, pois seria uma estrutura difícil de criar e gerir. O crescimento da área de desenvolvimento de software em conjunto com o amadurecimento das técnicas e linguagens de programação, a evolução do segmento relacionado aos dispositivos sensores também contribuíram para o sucesso desse movimento. É de extrema importância citar, também, os avanços nas áreas de sistemas embarcados e circuitos integrados, responsáveis por fornecer poder computacional para os dispositivos, reduzindo as dimensões e os custos de produção. O encontro de todas essas tecnologias foi o que proporcionou a realização da visão de Weiser. O maior desafio é fazer com que todas essas peças se encaixem em harmonia, formando um ambiente estável e seguro, capaz de promover o encontro do mundo real com o mundo virtual. Assim sendo, percebemos que a Internet das Coisas é a interseção de várias tecnologias trabalhando para fornecer autonomia de diversos sistemas, facilitando a vida do homem, que antes era peça importante para a execução dos processos e agora será capaz de definir os limites que deseja, deixando as máquinas realizarem o trabalho, e ao mesmo tempo ter condições de ajustarem as ações de acordo com o contexto em que estão inseridas.

2.1.1. Elementos da Internet das Coisas

Al-Fuqaha, em seu trabalho “Internet of Things: A Survey on Enabling Technologies, Protocols and Applications”(2015), faz uma análise das tecnologias que habilitam a IoT, e define os seis blocos que definem a IoT(figura 2). Compreendendo os blocos de construção da Internet das Coisas tem-se uma percepção maior do verdadeiro significado e funcionalidade da IoT.

- **Comunicação:** Neste bloco estão inseridas as tecnologias utilizadas para criar o canal de comunicação entre os objetos inteligentes, mas para atender a demanda deste novo paradigma, as tecnologias utilizadas devem levar em consideração as características do ambiente IoT, devem possuir um alto grau de confiabilidade, pois irão trabalhar em links com consideráveis taxas de erro e de ruído. Outra preocupação é que essas tecnologias tenham a capacidade de operar com um baixo consumo de energia, pois encontraremos pelo caminho dezenas de dispositivos alimentados por bateria, criando restrições de alimentação. Sem falar que a expectativa é que o número de aparelhos conectados chegue na casa dos 50 bilhões em alguns anos, imagina a demanda energética que será criada. Então, o consumo eficiente das fontes de energia torna-se um tópico importante que tem que ser levado em consideração nesse bloco. Protocolos como IEEE 802.11, Bluetooth, ZigBee que é um protocolo baseado nas especificações do IEEE 802.15.4, que veio, justamente para padronizar técnicas para construir redes de baixo consumo energético, confiáveis e escaláveis, são exemplos de tecnologias que, atualmente, são utilizadas para promover a comunicação entre objetos inteligentes.
- **Identificação:** O bloco de identificação é um dos principais pilares para o alcançarmos o pleno funcionamento de uma rede IoT e o encaixe perfeito de demandas e aplicações. Foi dito anteriormente que os dispositivos devem ser capazes de obter informações sobre o ambiente e de reconhecer outros aparelhos ao seu redor, para que isso seja feito sem maiores problemas é necessário que se tenha um método de identificação que torne cada objeto único no seu meio. É necessário, também, a correta diferenciação da identificação do objeto do seu endereço[Al-Fuqaha et al. 2015], para entender melhor podemos dizer que o ID representará o nome do objeto e o endereçamento indicará sua localização na rede. É uma área que demanda um esforço enorme de criação e gestão, levando em consideração o número de elementos e o problema que pode ser gerado se você identificar um objeto errado ou atribuir o mesmo endereço para aparelhos distintos. Como exemplo do que já é utilizado para esse bloco, temos o EPC, ou Código Eletrônico de Produto, que é uma chave utilizada para a identificação de produtos em uma cadeia de suprimentos [GS1BR.ORG,2017]. Tem também o Sistema de identificação numérica *Ucode*, que foi concebido para identificar

lugares e objetos do mundo real de maneira única, gerando uma tag que pode ser representada por um código de barras ou tag RFID. Para o endereçamento vamos encontrar tecnologias já utilizadas como o IPv4 e o IPv6, mas também vão surgindo novas implementações voltadas para as mesmas carências presentes na IoT, como é o caso 6LoWPAN que serve para dar suporte para as redes de baixo consumo, IEEE 812.15.4, para que essas possam trabalhar baseadas no endereçamento IPv6, tecnologia que oferece um range maior de endereços, bem superior ao seu antecessor, IPv4.

- **Sensores/Atuadores:** Na área de sensores serão encontrados os responsáveis por coletar dados e enviar para a nuvem ou banco de dados. A partir daí, esses dados serão analisados para fornecer informações relevantes para tomada de decisões. Os objetos presentes na rede são sensíveis ao contexto, são os sensores ou atuadores que realizam a interligação da rede com o meio real, formando o canal de entrada e saída, o ponto de encontro do mundo virtual e o real. Esses sensores podem ser desprovidos de capacidade de processamento, mas em boa parte das vezes vêm acompanhados de dispositivos como arduino ou raspberry-PI que fornece um canal de gerenciamento e controle para os clientes.
- **Serviços:** Al-Fuqaha, categoriza em seu trabalho quatro tipos de serviço, pois é assim que as funcionalidades são apresentadas em IoT.
 - *Serviço de identificação:* O serviço de identificação é o mais básico de todos, pois é utilizado por outros tipos de serviços. Qualquer aplicação necessita conhecer os objetos do mundo real, é através dessa identificação que eles são diferenciados e representados no mundo virtual.
 - *Serviços de agregação de informação:* Essa classe de serviço realiza a coleta e sumariza os dados brutos que necessitam ser processados antes de enviá-los a aplicação.
 - *Serviços de Colaboração:* Os serviços oferecidos nesta categoria, rodam no topo dos serviços de agregação, pois recebem os dados processados e tomam decisões ou reagem em conformidade com as informações consolidadas e as inclinações das aplicações.
 - *Serviços Ubíquos:* É a classe responsável por garantir que os serviços de colaboração estejam disponíveis a qualquer tempo.
- **Semântica:** Este elemento da IoT, refere-se aos métodos utilizados para realizar a padronização da apresentação das informações. Já é do nosso conhecimento que a quantidade de dispositivos que serão habilitados a conectarem-se à internet será muito maior no futuro, sabemos também que boa parte dos esforços aplicados no desenvolvimento de tecnologias para o crescimento da IoT é para promover a interoperabilidade entre os

diversos aparelhos que estarão em operação. Levando em consideração a gama de dispositivos presentes no mercado, o grande número de fabricantes e a atual falta de padrão internacional a ser seguido, podemos imaginar o quão grande é o desafio nesta parte. Podemos destacar algumas tecnologias de representação como o XML ou JSON, que servem para padronizar e organizar a estrutura em que os dados serão apresentados, bem como a forma de utilização.

- **Computação:** Nesse bloco estão representados os elementos que fornecem a capacidade computacional necessária para que simples “coisas” tornem-se smartobjects. Ao longo do tempo se observa a evolução dos computadores e o crescimento do poder computacional oferecidos por eles, depois vieram os telefones, passaram pelo mesmo ciclo de crescimento e evolução, podemos até considerar que ultrapassaram os computadores convencionais, pois podemos encontrar caso de pessoas que substituíram o PC ou notebook pelo smartphone, pois conseguiam realizar as mesmas atividades neste que são menores e trazem maior comodidade no seu uso. No entanto, nesta nova revolução começam aparecer aparelhos menos convencionais para se conectarem a internet. Na IoT, objetos como fogões geladeiras, lâmpadas e até mesmo banheiras, começam a oferecer serviços que podem ser controlados remotamente. Para isso devem possuir o mínimo de capacidade de processamento, e é nesse sentido que vem a contribuição de microcontroladores, FBGAs e outros elementos capazes de inserir uma inteligência em aparelhos como geladeiras ou fogões. Também estão inseridos neste bloco, os software que realizam o controle das aplicações. A junção desses elementos formam o “cérebro” da IoT [Al-Fuqaha et al. 2015], e em cima do funcionamento dessas peças, torna-se viável fazer com que os objetos equipados com essa capacidade consigam se comunicar com outros dispositivos, serem controlados de forma remota e que possam fornecer serviços aos seus usuários, na figura 2 temos a representação da junção de cada bloco para formar a IoT.

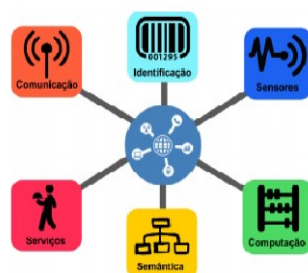


Figura 2 – Elementos de uma rede IoT(Fonte: SANTOS,2016)

2.1.2. Modelo básico de Arquitetura para Internet das Coisas em camadas

A Internet das Coisas se apresenta como o futuro, mas para que isso aconteça ainda é necessário vencer alguns desafios, como por exemplo, fornecer uma infraestrutura capaz de suportar o tráfego gerado pela enorme quantidade de dispositivos e capaz, também, de armazenar a grande massa de dados gerados. Surge como desafio, também, promover a interoperabilidade entre dispositivos de diversos fabricantes, escalabilidade, confiabilidade e segurança. A possibilidade de que bilhões de objetos inteligentes, cada qual com suas características, criando a necessidade de uma arquitetura flexível que suporte as diversas tecnologias utilizadas em um cenário de Internet das Coisas [SANTOS,2016]. Para determinar uma arquitetura que mostrasse a Internet das Coisas em todos os seus aspectos, se entraria em uma atividade muito complexa, então por questões de didática, optamos por utilizar um modelo básico apresentado por Bruno Pereira dos Santos, constituído de 3 camadas. E de forma sucinta, conseguir apresentar as 3 principais abordagens para ativação da IoT. O modelo é formado pelas camadas de **Aplicação**, **Redes** e **Percepção**, conforme é representado abaixo(figura 3).

2.1.2.1. Camada de Percepção

Nessa camada está concentrado o conjunto de dispositivos inteligentes e redes de sensores que realizam a coleta e geração de dados sobre os ambientes em que estão inseridos. Nesta camada se encontra os sensores e atuadores, responsáveis por fazer a interface do mundo físico com o digital, transformando em dados digitais, grandezas como umidade, temperatura, movimento, localização, velocidade, consumo de energia, quantidade de pessoas que entraram em um prédio, número de vagas livres em um estacionamento. Não podemos nos atentar somente ao fluxo do mundo real para o digital, pois a entrada de dados é tão importante quanto o resultado que elas irão influenciar. É na camada de percepção que encontraremos os atuadores, realizando o trabalho de controle dos estados e ações sobre as “coisas”. Contaremos, também, com a presença de etiquetas e leitores RFID, utilizadas para identificar objetos. Resumidamente, na camada de percepção teremos as tecnologias responsáveis por fazer o intermédio do mundo virtual e real, automatizando o controle de aparelhos e habilitando que suas funções sejam acionadas sem a intervenção direta do usuário daquele serviço.

2.1.2.2. Camada de Rede

É pré-requisito para o funcionamento da IoT, que os vários dispositivos espalhados pelo ambiente tenham a capacidade de perceber e de se comunicarem com os dispositivos vizinhos Na camada de rede que é aonde se encontra as tecnologias necessárias para estabelecer os canais de

comunicação entre os objetos participantes de determinado ambiente. Uma grande parte dos serviços oferecidos em um ambiente IoT, dependem de uma intensa análise de dados, o que implica em grande demanda poder de processamento e consequentemente altas taxas de consumo energético, como é sabido, essas são características opostas a maioria dos objetos, fazendo que a maioria dos serviços sejam hospedados na nuvem ou em computadores centralizados com maior capacidade de processamento, logo, a camada de rede será responsável por promover a ligação entre o ambiente e o local onde as aplicações estão hospedadas, tanto para permitir o envio dos dados gerados na camada de aplicação, bem como a interação entre estes ambientes, geralmente separados geograficamente.

2.1.2.3. Camada de Aplicação

A camada de aplicação é a responsável por prover os serviços, atendendo as requisições de clientes e promovendo a interoperabilidade entre os sistemas de uma rede povoada de objetos inteligentes, Pode ser considerada como a mais importante, pois nela é concentrada a inteligência capaz de tornar real os serviços oferecidos pela Internet das Coisas, que é caracterizada por possuir diversos dispositivos imersos no ambiente, capturando informações, atuando nos ambientes e comunicando-se via internet para executar processos complexos que darão respostas diferentes, influenciados pelos estímulos e dados absorvidos. Na camada de aplicação serão implementados programas com as regras de negócio que definem os serviços oferecidos, aplicativos para realizar a interface entre usuário e dispositivos, dado o número de aparelhos que podem atender um mesmo usuário, logicamente, este necessitará de uma forma eficiente de gerenciar e acionar cada dispositivo. Também encontraremos nesse nível da arquitetura, estruturas conhecidas como *middleware*, ficam entre a camada de Rede e de Aplicação e objetivam garantir a interoperabilidade entre os milhares e heterogêneos dispositivos, facilitar o gerenciamento e padronizar as formas de acesso a informação[Paulo F. et al,2015].



Figura 3 - Modelo básico de 3 camadas para Internet das Coisas(SANTOS, 2016)

2.2. Tecnologias Relacionadas

Nesta seção iremos apresentar as principais tecnologias que, atualmente, são utilizadas na construção da Internet das Coisas.

2.2.1. Tecnologias de Rede

Podemos depositar na camada de rede uma grande parcela de importância para que a IoT viesse a torna-se realidade. As evoluções das tecnologias de rede vieram e solucionaram o problema que se tinham para conectar todos os dispositivos, sensores e atuadores e prover um modo de comunicação confiável. O número de tecnologias que podem atender as demandas geradas pela Internet das Coisas é enorme. Nesta seção segue abaixo o resumo de alguns protocolos já utilizados e que tentam ganhar seu espaço como padrão.

2.2.1.1. Ethernet

O Padrão Ethernet nasceu nos laboratórios da empresa Xerox, em Palo Alto. Criado por Robert Metcalf, e rapidamente tornou-se uma das tecnologias mais utilizadas para comunicação em redes locais. A tecnologia enfrentou dificuldades para se estabelecer como padrão devido a falta de padronização, entretanto a norma IEEE 802.3 foi oficializado em 1983 e tornou-se padrão para as camadas de enlace e física de redes locais, ganhando o espaço e dividindo o mercado com grandes tecnologias como Token Ring e ARCNET. A norma que define o padrão Ethernet, basicamente, trabalha nas especificações de 3 elementos: meio físico, controle de acesso ao meio e formato do quadro ethernet. O protocolo Ethernet parte do princípio de que todas as máquinas estão conectadas na mesma linha de transmissão, e a comunicação é baseada no protocolo de acesso ao meio CSMA/CD(Carrier Sense Multiple Access with Collision Detection).

10Base-T Ethernet (IEEE 802.3)

Foi a primeira normatização do padrão Ethernet. Entre suas principais características, podemos destacar a velocidade em que o sistema atua, sendo ela uma taxa de 10Mbps. Outro ponto importante é a limitação física que determina uma distância máxima de 100 metros entre o terminal do usuário e o switch na outra ponta, responsável por separar os domínios de colisão.

Fast Ethernet (IEEE 802.3u)

Evolução do padrão Ethernet, atingindo uma velocidade 10 vezes maior que a especificação anterior, sendo agora de 100 Mbps. Este aumento foi possível, devido ao módulo de suporte full-duplex inserido nesta nova norma, tornando possível que um equipamento transmita e receba dados ao mesmo tempo, fazendo uso de dois pares de fio, de um cabo de par trançado categoria 5, sendo um para download e outro para upload. Existem diversos modos de implementação da FastEthernet, podendo ter como meio físico, o cabo de cobre: 100BASE-TX, 100BASE-T4, 100BASE-T2, ou fibra ótica: 100BASE-FX, 100BASE-SX, 100BASE-BX. Sendo o modelo 100Base-TX o mais utilizado.

Gigabit Ethernet (IEEE 802.3z/IEEE 802.3ab)

Neste padrão que começou a ser desenvolvido em 1997, se chegou a velocidade de 1 Gbps. Durante seu desenvolvimento, a norma acabou dividindo-se em 4 padrões distintos: 1000baseLX, 1000baseSX, 1000baseCX e o 1000baseT, sendo todos definidos pela norma IEEE 802.3z, exceto o Modelo 1000base-T que é definido pela norma IEEE 802.3ab. Sendo este último modelo o mais aceito pelas empresas, pois, respeitando a distância limite de 100 metros, o modelo suporta que seja construído utilizando cabos categoria 5, reaproveitando o cabeamento feito para FastEthernet. Ao contrário do modelo 100Base-TX, que para funcionar, necessita de cabos categoria 6, tornando necessário a troca do cabeamento existente, medida que não agradou os empresários.

10 Gigabit Ethernet (IEEE 802.3ae)

Padrão Ethernet criado para atingir a velocidade de 10 Gbps. Devido a sua característica de poder operar em longas distâncias, o padrão 10GibitEthernet vem sendo amplamente aplicado em redes LAN, MAN e WAN.

2.2.1.2. IEEE 802.11 (Wi-Fi)

O Wi-Fi é a tecnologia de comunicação mais popular, pois ela está presente em quase todos os lugares, fazendo parte do cotidiano das casas, escritórios, indústrias e até espaços públicos das cidades. O padrão IEEE 802.11 (Wi-Fi) define um conjunto de padrões de transmissão e codificação, que hoje é amplamente aceito como padrão para comunicação sem fio, baseado na divisão da área coberta em células denominadas BSA(Basic Service Area) que podem variar de tamanho de acordo com o ambiente e com as características dos equipamentos transmissores/receptores [GARCIA,Luis]. O modo de transmissão do padrão 802.11 é todo baseado na utilização de radiofrequência para enviar dados e utiliza duas técnicas para tal processo, o DSSS, FHSS e OFDM, trabalhando com a codificação da informação e variando as maneiras de modulação do sinal, buscando equilibrar três variáveis: velocidade, capacidade e distância de transmissão.[GARCIA, Luis]. Depois da aprovação desse padrão no ano de 1997 que atingia taxas de transmissão entre 1 e 2 Mbps, surgiram outras versões que buscavam superar a primeira versão.

A seguir se fala das versões mais utilizadas que são 802.11a, 802.11b, 802.11g, 802.11n e a 802.11ac. A primeira variação da proposta original foi o padrão 802.11b, assim como o seu antecessor, trabalha em torno da frequência de 2,4 Ghz, mas com uma proposta para oferecer uma velocidade de 11 Mbps. No mesmo ano, 1999, vem o padrão 802.11a que acrescenta modificações mais contrastantes como os modelos anteriores, neste momento é incluído um novo método de transmissão que é o OFDM, Orthogonal Frequency-Division Multiplexing, e altera a frequência de trabalho que vai para 5 Ghz, este modelo permite que as redes alcancem velocidades de 54 Mbps.

O modelo seguinte é a versão 802.11g que era compatível com os 802.11b e oferecia a mesma velocidade de 54Mbps da versão anterior. A próxima evolução fica por conta do padrão 802.11n que implementa o conceito de MIMO, e operando a uma velocidade de 300Mbps.

2.2.1.3. Protocolo ZigBee

O padrão ZigBee é baseado na especificação IEEE 802.15.4 para a camada de enlace e física. As principais características do ZigBee são a baixa vazão, reduzido consumo energético, e baixo custo. O ZigBee opera na frequência 2.4-GHz ISM, mas é capaz de operar em outras duas frequências, 868 MHz ou 915 MHz ISM. Esta tecnologia pode alcançar a vazão de 250Kbps, mas na prática somente taxas inferiores são alcançáveis. O ZigBee também permite que os dispositivos entrem em modo sleep por longos intervalos de tempo para economizar o consumo de energia, fazendo com que a vida útil da bateria seja prolongada. Atualmente, novos dispositivos ZigBee permitem o uso técnicas de colheita de energia do ambiente (Energy Harvesting, discutida mais adiante) para diminuir o uso da bateria por operações [Reiter 2014]. Em uma rede ZigBee podemos categorizar os elementos em três classes, o coordenador, responsável por iniciar a rede e definir o canal de comunicação, gerenciar os nós e armazenar informações da rede, temos os roteadores responsáveis pelo encaminhamento das mensagens entre os nós, a terceira classe representa os “end devices”, nós que já caracterizam os dispositivos existentes de um ambiente IoT. É importante, também, ressaltar as camadas que compõem um dispositivo ZigBee, simples de ser implementada, A pilha completa não excede 32 kBytes[FRIAS R,2004].

2.2.1.4. Bluetooth Low Energy

Bluetooth é um protocolo de comunicação inicialmente criado pela Ericsson para substituir a comunicação serial RS-232 [1]. Atualmente, o órgão Bluetooth Special Interest Group (SIG) é responsável por criar, testar e manter a tecnologia. Além disso, Bluetooth é uma das principais tecnologias de rede sem fio para Personal Area Networks PANs, que é utilizada em smartphones, headsets, PCs e outros. O Bluetooth se divide em: Bluetooth Clássico Basic Rate/Enhanced Data

Rate (BR/EDR), que são as versões 2.0 ou inferiores; Bluetooth High Speed (HS), versão 3.0; e o Bluetooth Low Energy (BLE), versão 4.0 ou superior. As versões mais antigas do Bluetooth, focadas em aumentar a vazão, tornou o protocolo mais complexo e, por consequência, não otimizado para dispositivos com limitações energéticas. Ao contrário das versões anteriores, o BLE possui especificação voltada para baixo consumo de energia, permitindo dispositivos que usam baterias do tamanho de moedas (coin cell batteries) prolonguem a sua vida útil, ao passo que mantém comunicação.

2.2.1.5. 3G/4G

Os padrões Celular 3G/4G podem também ser aplicados no cenário da IoT. Projetos que precisam alcançar grandes distâncias podem aproveitar as redes 3G/4G. Por outro lado, o consumo energético da tecnologia 3G/4G é alto em comparação com outras tecnologias. Porém, a sua utilização em locais afastados e baixa mobilidade podem compensar esse gasto. No Brasil, a frequência utilizada para o 3G 1900MHz e 2100MHz (UMTS), já o padrão 4G (LTE) utiliza a frequência 2500MHz. A vazão de dados alcançada no padrão 3G é de cerca de 2Mbps e no padrão 4G pode alcançar até 150Mbps [2].

2.2.1.6. LoRaWan

O Long RangeWide Area Network, foi projetado e otimizado para prover redes de longa distância (Wide Area Networks (WANs)). Como principais características se tem o baixo custo, a mobilidade, e segurança na comunicação para IoT. Além disso, o padrão oferece suporte a IPv6, a adaptação 6LoWPAN e opera sobre a topologia estrela [3]. O fator atrativo do LoRAWAN é o seu baixo custo e a quantidade de companhias de hardware que estão adotando. A vazão de dados alcançada desta tecnologia alcança valores entre 0.3kbps a 50kbps. O consumo de energia da LoRaWan é considerada pequena, o que permite que os dispositivos mantenham-se ativos por longos períodos. A LoRaWANs utiliza a frequência ISM sub-GHz, o que permite que as ondas penetrem grandes estruturas e superfícies com raio de 2km a 5km em meio urbano e 45km no meio rural. Os valores de frequência mais usadas pelo LoRaWan são: 109 MHz, 433 MHz, 866 MHz, 915 MHz. O MTU adotado pelo padrão LoRaWAN é de 256 bytes [4].

Sigfox O padrão SigFox utiliza a tecnologia Ultra Narrow Band (UNB) que foi projetada para lidar com pequenas taxas de transferência dados. O padrão ainda é bastante recente, e já possui uma larga cobertura que abrange dezenas de milhares de dispositivos, os quais estão espalhados na Europa e na América do Norte. A SigFox atua como uma operadora para IoT, com suporte a uma série de dispositivos. A principal função é abstrair dificuldades de conexão e prover uma API para

que os usuários implementem sistemas IoT com maior facilidade. O raio de cobertura oficialmente relatada, em zonas urbanas, está entre 3km e 10km e em zonas rurais entre 30km a 50km. A vazão de dados varia entre 10bps e 1000bps. O MTU utilizado é de 96 bytes. O SigFox possui baixo consumo de energia e opera na faixa de 900MHz [5].

2.2.2. Protocolos da Camada de Aplicação

Em um ambiente IoT, um dispositivo deve ser capazes de informar quais serviços ele é capaz de fornecer, seu estado atual, sua identificação, enfim, uma série de informações que necessitam ser trocadas durante todo o tempo. A integração entre dispositivos e aplicações na troca de informações torna-se processo imprescindível para a implementação de um rede IoT. O padrão de mensageria, inicialmente pensada por Vivek Ranadivé[CREATE SOLUTIONS,2012], permite que as aplicações atinjam um grau maior de maturidade quando o assunto é escalabilidade e estabilidade, bem como torna mais fácil e confiável a integração entre sistemas completamente diferentes [DEVMEDIA].

Ranadivé, vislumbrou a idéia de que deveria existir um barramento de software baseado no mesmo princípio do barramento entre a placa mãe e outros dispositivos do PC. Começou a desenvolver a aplicação TIB, *The information Bus*, e em 1985 teve seu primeiro cliente Goldman Sachs, o conceito aplicado por Ranadivé foi muito bem aceito no mercado e a partir daí muitos outros empresas começaram a desenvolver seus sistemas de mensageria, entretanto a comunicação entre essas novas aplicações era complicada e na maioria da vezes dificultada pelos fabricantes[CREATIVE SOLUTIONS].

Posteriormente, nasceram algumas tentativas de apresentar um padrão para vencer essa o problemas de comunicação e que existisse uma compatibilidade entre os softwares. Nesta seção iremos abordar os principais protocolos de mensageria que atualmente estão no mercado, que buscam o mesmo objetivo, porém apresentam diferenças em alguns aspectos, que fazem com que se tenha que conhecer esses detalhes para melhor escolher qual método utilizar na nossa aplicação.

2.2.2.1. Advanced Message Queuing Protocol (AMQP)

O AMQP é um padrão aberto, nascido da iniciativa do JPMorgan Chase & Co em parceria com a iMatix, e tem a finalidade de passar mensagens de negócios entre aplicativos ou organizações. Segundo a própria organização formada por empresas do setor financeiro e empresas de TI, o AMQP é capaz de conectar sistemas de diferentes organizações e diferentes plataformas. Pensado também para conectar através do tempo, logo, não é necessário que os dois sistemas estejam simultaneamente ativos, e de forma com que pudesse trabalhar com links de comunicação de baixa qualidade[AMQP.ORG]. Projetado para ser um padrão confiável, seguro, completo e que

garantissem a interoperabilidade dos sistemas. Em sua camada mais baixa, é definido um protocolo binário para transportar, de maneira eficiente, mensagens através da rede entre dois pontos. Em seguida vem a camada de mensagem definindo um formato padrão de mensagem, e para que o processo consiga ser capaz de se comunicar utilizando esse protocolo, ele deve ser capaz de enviar e receber mensagens codificadas nesse padrão.

O foco principal do AMQP é não perder mensagens e manter a ordem de envio e recepção das mensagens. Para atingir esse objetivo, ele define um protocolo de transporte *peer-to-peer* que trabalha sobre TCP e executa um protocolo de confirmações de mensagens que garante a entrega das mesmas independentemente de falhas ou reinicialização do sistema.

eXtensible Messaging and Presence Protocol (XMPP): Foi inicialmente desenvolvido como um protocolo para mensagem instantânea para conectar pessoas através de mensagens de texto. Esse protocolo utiliza XML para suas mensagens e é executado sobre o TCP. Um dos pontos chave do XMPP é a utilização de `name@domain.com` como esquema de endereçamento, oferecendo um jeito fácil de endereçar um dispositivo IoT na Internet através da infraestrutura existente de DNS.

Message Queue Telemetry Transport - Sensor Network (MQTT-SN): *O protocolo MQTT-SN é uma variação do protocolo MQTT para redes que não sejam baseadas em TCP/IP. O protocolo MQTT-SN tem sua PDU encapsulada pelo protocolo UDP.*

Constrained Application Protocol (CoAP): É um padrão que adapta os conceitos do HTTP para dispositivos com restrições computacionais e de largura de banda, e segue um modelo cliente/servidor, ele é baseado no modelo arquitetural REST, no CoAP os servidores disponibilizam recursos por meio de um URL e os clientes acessam esses recursos usando métodos pré-definidos como GET, PUT, POST e DELETE. Como, tanto o HTTP, quanto o CoAP usam o modelo REST, eles podem ser facilmente integrados usando proxies, de forma que, por exemplo, uma aplicação cliente pode nem ficar sabendo que está acessando um sensor. Por ter sido projetado para dispositivos de baixo custo e poder computacional, o CoAP emprega uma pilha de protocolos “enxuta”, baseada em UDP sobre IP, ou sobre 6LoWPAN. O cabeçalho CoAP possui um tamanho fixo de 4 bytes e suas mensagens causam nenhuma, ou muito pouca, fragmentação na camada de enlace. Em relação aos aspectos de segurança, o CoAP optou por usar DTLS (Datagram Transport Layer Security) que é equivalente ao uso de chaves RSA de 3072 bits, o que lhe confere uma excelente segurança com pouco consumo computacional (se comparado com outros métodos).

HTTP REST: é um conjunto de princípios que definem como Web Standards como HTTP e URIs devem ser usados. O REST é um estilo de arquitetura para o desenvolvimento de aplicações distribuídas e é a base de como a Web está atualmente organizada. Basicamente, REST permite a

criação de serviços fracamente acoplados, que podem ser facilmente reusados, usando tecnologias como URIs, HTTP e padrões de tipos de dados. Na prática, isso significa que é possível interagir com objetos via navegadores Web da mesma forma que se surfa na Web. Assim, por exemplo, dados podem ser coletados de sensores localizados em qualquer lugar do mundo, através de seus URLs, processados e visualizados em navegadores Web usando HTML, CSS e JavaScript.

2.2.3. Segurança

Quando se fala de segurança, se esta falando dos riscos e ameaças que estão sujeitos ao realizar qualquer atividade, dos modos que temos de nos resguardar frente a esses obstáculos e da busca da melhor relação entre custos, eficácia e necessidade desses métodos. Atualmente, com a evolução tecnológica e a redução do custo de recursos de armazenamento e comunicação O mercado tem trabalhado cada vez mais, sendo suportadas por essas tecnologias, tornando a segurança da informação ponto importante. Os sistemas são construídos sempre obedecendo os princípios mais importantes dessa área, sendo eles: confidencialidade, integridade, disponibilidade e autenticidade. Devemos ter em mente que segurança é organizada em camadas em que o comprometimento de camadas inferiores coloca em risco as superiores, portanto não adianta utilizar as melhores práticas no meio sendo que a camada mais baixa está fragilizada.

2.2.3.1. Segurança em um ambiente IoT

Segurança sempre foi um quesito importante, porém em ambientes IoT ganham um peso maior, tendo em vista que a maioria dos dispositivos possuem limitações de hardware e de software, podem estar localizados em locais de difícil acesso ou que não permitem realizar o controle adequado. O grande número de dispositivos conectados a rede, também é um fator agravante para que montar uma rede com o mínimo de segurança necessário. Neste momento já temos milhares de dispositivos conectados a internet, mas o que percebemos é a não preocupação dos fabricantes com o quesito segurança ao lançar um aparelho que pode conectar-se a rede. A prioridade tem sido o baixo custo para criação desses eletrônicos, os departamentos de pesquisa e desenvolvimento buscam aprimoramento do objeto no que se diz respeito a sua funcionalidade, mas o quesito segurança geralmente é negligenciado e não tem tido a importância que merece[HOEPERS,2016].

Devido as suas características um ambiente IoT já é mais vulnerável, pois boa parte dos seus canais de comunicação são baseados nos meios de comunicação sem fio. A comunicação feita por meio de ondas eletromagnéticas torna mais complicada a missão de limitar quem pode ter acesso ao meio ou não, pois pessoas não autorizadas que estão dentro dos limites de detecção da rede poderão capturar as dados que ali transitam[UFRGS,2017].

Vulnerabilidades: Em um ambiente IoT cada objeto conectado é um ponto de fragilidade da rede. É onde se apresenta o maior número de elementos que podem ser acessados sem senha de acesso, criptografia fraca para guardas as informações contidas no objeto.

	Vulnerabilidades
Projetos sem levar em consideração a segurança	P&D não vai se preocupar com segurança, A prioridade é o baixo custo tanto de <i>hardware</i> , quanto de <i>software</i>
Defeitos de software	Já tivemos casos como um carro da Fiat <i>Chrysler</i> , foi controlado remotamente. Os invasores utilizaram vulnerabilidades do aplicativo de entretenimento <i>Uconnect</i> .
Falhas na configuração	Equipamentos saiam de fábrica permitindo o acesso com privilégios de usuário root por padrão. Equipamentos que permitem a atualização de software sem a verificação de autenticidade e origem .
Uso inadequado	Usuários instalam equipamentos, porém configuram senhas fracas ou não colocam senhas
Fraquezas advindas da complexidade do sistema	Os sistemas são formados por junções de várias tecnologias, logo a fraqueza de outros sistemas, como por exemplo aplicações web, que podem fornecer <i>exploits</i> , fragilizando a estrutura como um todo.

Tabela 1 - Vulnerabilidades da Internet das Coisas

Dessa forma os sistemas estão sujeitos a sérios riscos de indisponibilidade de serviço, perda de privacidade, furto de dados, perdas financeiras, danos a imagem e perda de confiança na tecnologia [HOEPERS,2016].

2.2.3.2. Algoritmos de criptografia leve

A medida que o tempo passa, mais e mais dispositivos se conectam as redes e passam a atuar diretamente ou indiretamente, nas mais diversas atividades, podendo ser elas simples como regular o horário e acionar as lâmpadas do jardim, ou administrar os remédios dos nossos avós. Se um dispositivo desse é comprometido, os danos causados podem ser mínimos, como também podem ser desastrosos. Esses elementos, em sua grande maioria, apresentam limitações que dificultam a implementação das técnicas de segurança já existentes. Por esse motivo, vem sendo estudado e desenvolvido, técnicas de criptografia leve que, resumidamente, buscam fornecer algoritmos para que seja possível implementar em ambientes restritos a criptografia, na intenção de fornecer segurança. A construção desses algoritmos baseia-se em duas visões: software e hardware.

Dentre as limitações de software que podemos enfrentar, podemos destacar a limitação de memória e capacidade de processamento. Desta forma a métrica mais observada é o tamanho do código. Do ponto de vista do hardware, podemos encontrar como obstáculo o consumo energético. Quanto mais rápido o código é executado, menor é o consumo de energia, pois assim o quanto antes o aparelho voltar para o *sleep mode*, menor será a energia consumida. Devemos levar em consideração também as limitações de poder computacional, que é reduzido nesses dispositivos devido as suas limitações físicas. Para alcançar um bom desempenho o código deve ser otimizado para entregar os resultados mais rápido. Existem vários algoritmos de encriptação existentes, que podem ser separados em simétricos e assimétricos. Onde os códigos simétricos são os que utilizam uma única chave para criptografar e descriptografar uma mensagem, diferente do código assimétrico que possui duas chaves, pública e privada, sendo a primeira utilizada para criptografar e a segunda para abrir as mensagens. Mas vamos falar de 2 métodos de cifrar uma mensagem que são amplamente utilizados em redes de dispositivos restritos, o DES e AES. São códigos simétricos, onde o DES utiliza uma chave de 64 bits e o AES pode usar chaves de 128, 192 e 256 bits.

Data Encryption Standard (DES): O algoritmo DES visa prover um método padrão para criptografar e assim proteger as informações que não podem ser conhecidas por qualquer um. Esse método é considerado simétrico, pois utiliza a mesma chave para cifrar e decifrar as mensagens. O método DES cifra mensagens em blocos de 64 bits utilizando uma chave do mesmo comprimento, entretanto 8 bits são para verificar paridade e são removidos, fazendo com que o tamanho real da chave seja de 56 bits, gerando uma mensagem de comprimento de 64 bits.

Advanced Encryption Standard(AES): O Algoritmo AES nasceu de um concurso organizado pelo NIST(Instituto Nacional de Padrões e Tecnologia dos EUA). O concurso fazia algumas exigências, que deveriam ser cumpridas como pré-requisitos para que o código pudesse participar. Os passos submetidos ao concurso, deveriam superar em desempenho o seu antecessor o DES e 3DES, deveria utilizar chaves de 128, 192 e 256 bits e cifrar blocos de 128 bits. O AES realiza o trabalho de criptografar os dados em 10, 12 ou 14 rodadas, dependendo do tamanho da chave. O AES se destaca, pois, se comparado com outros algoritmos ele roda mais rapidamente. Esse ganho de desempenho pode ser atribuído ao fato de que algumas operações podem ser realizadas em paralelo, agilizando a conclusão do algoritmo. Algumas características da implementação do código fazem com que ele tem algumas vantagens. Utiliza pouco código e memória. Por não utilizar operações aritméticas, não exige capacidade elevada de processamento. Fazendo com que ele seja uma ótima escolha para trabalhar em pequenos dispositivos e restrições de memória e/ou capacidade de processamento.

2.2.3.3. Protocolos de Segurança

Sabemos que a Internet das Coisas tornou-se possível após a junção de diversas tecnologias, na área de segurança as estruturas se constroem de maneira análoga, a partir de tecnologias já consolidadas, sendo utilizadas de forma a prover um ambiente seguro, respeitando as limitações imposta pelas características de um ambiente IoT. Neste trabalho iremos falar do protocolo TLS, por ser amplamente utilizado em conjunto com o MQTT.

Transport Layer Security (TLS): O protocolo TLS foi projetado para prover a privacidade na troca de dados entre aplicações de comunicação e garantir a integridade dos dados na transmissão. Formado por duas camadas, o TLS é dividido em TLS Record Protocol e TLS Handshake Protocol. O protocolo TLS Record é utilizado para construir conexões seguras com duas propriedades, conexão privada: Utiliza criptografia simétrica para encriptar os dados e as chaves são geradas exclusivamente para cada conexão por outro protocolo, como por exemplo o TLS Handshake. A segunda propriedade se baseia na confiabilidade da conexão, realizando o transporte de mensagem, valendo-se de um autenticador de mensagens, que possibilita verificar tanto a integridade da mensagem, quanto a autenticidade da mesma. O protocolo TLS Handshake, funciona de outra maneira, ele possibilita que o servidor e o cliente autentique a identidade um do outro e negocie chaves de criptografia e os algoritmos utilizados, antes de iniciar a transmissão. A conexão segura oferecida pelo TLS Handshake possui três propriedades: A autenticação da identidade dos pares pode ser feita por criptografia assimétrica. Segunda propriedade carrega a ideia de que a negociação de um segredo compartilhado é segura e não pode ser interceptada mesmo que o atacante esteja no meio da comunicação entre os pares. E por último, a conexão é confiável, dado que nenhum atacante poderá modificar as negociações de comunicação sem que as partes percebam.

Objetivos do protocolo TLS: Abaixo, elencar os objetivos do protocolo. Assim ficará evidente porque é um dos mais utilizados na área da IoT.

Segurança: O TLS tem como objetivo ser capaz de estabelecer uma conexão segura entre dois pares que desejam trocar dados e necessitam de um meio de comunicação seguro e livre de ameaças.

Interoperabilidade: Outras aplicações devem ser capazes de realizar troca de chaves criptográficas sem a necessidade de cada um conhecer o código do outro.

Extensibilidade: O TLS procurar ser o mais flexível possível, facilitando a entrada de novos códigos e algoritmos de criptografia, evitando a necessidade de criar um novo protocolo ou implementar uma nova biblioteca de segurança.

Eficiência Relativa: Para reduzir o consumo de banda na rede e o impacto da utilização de cpu ocasionado pelas operações criptográficas. O TLS incorporou o esquema de cache, salvando sessões e consequentemente, reduzindo o número de conexões criadas a partir do zero.

Podemos perceber que os três últimos itens vão de encontro com necessidades específicas da IoT.

2.2.3.4. Vulnerabilidades

Dentro de uma rede IoT, teremos milhares de dispositivos, dos mais variados tipos e fabricantes. A Internet da Coisas está começando sem uma padronização, o que por se só, representa ponto de vulnerabilidade. Entretanto, o ponto em questão é que a falta de padronização, transforma cada aparelho desse em um potencial ponto de ataque, tendo em vista que os fornecedores não seguem nenhuma norma para garantir um grau mínimo de qualidade ou segurança dos dispositivos.

Devido as limitações impostas pelas características de uma rede com força de alimentação e capacidade de processamento limitadas, se encontram problemas ao tentar cifrar as mensagens que tramitam em nossa rede, levando informações importantes. Os problemas enfrentados com criptografia, podem expor os dispositivos e usuários a ataques onde o sigilo dos dados é comprometido, e pessoas não autorizadas tomam posse dessas informações.

A segurança em sensores sem fio deve ser reforçada, visto que esses sensores lidam com informações privadas dos usuários. Contudo, existem obstáculos que dificultam o desenvolvimento de protocolos seguros, como reduzida memória, fonte de energia limitada e pouco espaço para código.

Kumari et al. (2010) apud De Jesus Junior e Moreno (2016) denotam que a comunicação entre os sensores não ocorre de forma confiável, seja pela fragilidade inerente ao meio de comunicação, seja pela possibilidade de conflitos, ou ainda, pela latência que dificulta a sincronização de chaves criptográficas entre os dispositivos. As principais ameaças de ataque físico são:

Subcamada física: Jamming

O ataque Jamming é caracterizado pela ação de neutralizar os dispositivos de transmissão. Para efetuar o ataque é necessário, conhecer a frequência da transmissão do dispositivo sem fio, com essa informação em mãos, é gerado um sinal capaz de interferir no sinal original, o que impede os dispositivos dentro do raio de alcance de receber ou transmitir mensagens. Os nós ficam isolados enquanto recebem a interferência. De modo a evitar essa interferência são recomendados saltos de frequência, porém é uma técnica não recomendada por exigir processamento extra e limitado alcance de frequências. A técnica Ultra Wide Band (UWB) é recomendada nesse caso já que

consome pouca energia. Consiste em mandar pequenos pulsos em nano segundos através da frequência, dificultando a identificação do sinal.

Ligação: Denial of Service (DoS)

Esse ataque é produzido por uma falha não intencional nos nós ou por ação maliciosa. Essa ação tenta sobrecarregar os nós mandando pacotes extras desnecessários e, assim, evita que os usuários da rede acessem os serviços. Esse ataque tenta diminuir a capacidade da rede de fornecer o serviço.

Rede - Selective Forwarding:

Um nó malicioso joga pacotes de seu interesse e encaminha pacotes seletivos para minimizar a suspeita para nós vizinhos. Esse ataque atrapalha o direcionamento das mensagens para os nós de rede certos. Existem dois meios de evitar que isso ocorra. Um deles é enviar dados usando vários caminhos. O outro é a detecção de nós comprometidos que tem comportamento ruim em termos de encaminhamento seletivo em busca de um caminho alternativo.

Black role/Sinkhole: um nó malicioso age como um buraco negro para atrair todo o tráfego do sensor de rede. O hacker escuta os pedidos de rota e sugere um caminho mais curto ao nó de destino. Esse ataque pode afetar até os nós mais distantes.

Hello Flood Attack: usa pacotes HELLO como arma para convencer os sensores em WSN. O hacker com um alto alcance de transmissão de radio e alto poder de processamento envia pacotes HELLO a uma serie de nós que são dispersos em uma grande área dentro de uma WSN. Os sensores são convencidos de que o adversário é seu vizinho. Como consequência, ao enviar a informação para a estação base, os nós alvo tentam atravessar o hacker pensando ser o vizinho e então são falsificados pelo hacker. A chave para combater esse ataque é autenticação. Um protocolo de transmissão autenticado é uma proposta eficiente, como, por exemplo, μ TESLA.

Wormhole: O hacker pega os pacotes (ou bits) em um local na rede de túneis e redireciona para outro local. A retransmissão de bits poderia ser feita de forma seletiva. O Wormhole é uma ameaça significativa para redes de sensores sem fio porque este tipo de ataque não requer comprometimento do sensor na rede, em vez disso, pode ser realizada mesmo na fase inicial, quando os sensores estão descobrindo os vizinhos.

Sybil Attack: em varias situações os sensores em wsn precisam trabalhar juntos para executar uma tarefa e assim distribuir sub tarefas e redundância de informações. Esse ataque tenta degradar a integridade da informação, da segurança e do uso de recursos do algoritmo. Pode ser usado para atacar a distribuição de armazenamento, mecanismo de roteamento, conjunto de dados, justa alocação de recursos e de detecção de mau comportamento. A verificação da identidade é o requisito fundamental para o combate do Sybil attack. Ao contrário das redes tradicionais, a verificação de identidade em WSN não pode ser feita com uma única chave simétrica e algoritmo partilhado de chave pública por causa da limitação computacional de WSN.

De transporte: Flooding Attack

Adversários tentam explorar os protocolos que mantêm o estado em cada extremidade da conexão. Por exemplo, o adversário envia muitas solicitações de estabelecimento da conexão para o nó alvo para esgotar os seus recursos. Uma solução contra este ataque é limitar o número de conexões que um nó pode fazer, o que pode impedir a conexão a nós legítimos da rede.

De aplicação: invasão e comprometimento dos dados da rede de sensores

Na camada de aplicação, os dados são recolhidos e geridos. Aqui, nós sensores podem ser subvertidos para revelar suas informações incluindo a divulgação de chaves criptográficas, portanto, comprometer toda a rede de sensores. Além disso, um nó pode ser comprometido a funcionar mal e gerar dados imprecisos e este efeito pode ser pior quando o nó é cluster leader no WSN.

2.2.4. Tecnologias de prototipagem de dispositivos

Nesta seção abordaremos um pouco do que temos disponível no mercado para a construção de objetos capazes de interagir com outras máquinas e com o ambiente em que está inserido. Com a evolução dessas tecnologias, conseguimos fazer com que objetos simples possam responder a estímulos analógicos e digitais [FERREIRA, 2014]. As placas para prototipagem de dispositivos eletrônicos possibilitam a elaboração de projetos de eletrônica e robótica, tratam-se de placas com processadores e dispositivos de entrada e saída. A característica mais importante é que essas placas são programáveis, logo, podem ser utilizadas em qualquer projeto. Abaixo falaremos das principais plataformas existentes no mercado: Arduino, Raspberry Pi, Galileo e Edison.

2.2.4.1. Arduino

O arduino nasceu no ano de 2005, com o objetivo de criar um dispositivo barato com fins educacionais. Tornou-se popular por sua versatilidade, atendendo aos mais diversos projetos e, também, por ser baseada no conceito de hardware livre, podendo ser personalizada. A placa está

mais voltada para engenheiros de hardware ou eletrônica, que gostam de criarem seus próprios dispositivos, este projeto é o que alavancou o uso do IoT, pois as compatibilidades com conexões externas são quase infinitas, porém existe um custo a mais no projeto, pois cada tipo de conexão vem em um módulo que é comprado separadamente, e comumente são conhecidos como Shields. Os programas que irão determinar como o Arduino trabalhará são escritos na linguagem de programação C/C++, e para que funcione corretamente, é necessário que os códigos sejam pequenos, pois a capacidade de armazenamento interno é limitada, são exatos 2K de RAM e 32K de Flash.

2.2.4.2 Raspberry PI 3

Criado no Reino Unido, também com objetivos educacionais como o menor computador do mundo (processador quad-core de 1,2 GHz), utilizando a tecnologia SoC (system on a chip), ou seja, todo o hardware em apenas uma única placa, tendo como objetivo ser um dispositivo para fins acadêmicos auxiliando no ensino de tecnologia, aprendendo a programar e desenvolver a sociedade. Sua compatibilidade com conexões externas através de GPIO (General Purpose Input/Output) dá capacidades de gerir outros dispositivos que fazem parte da Internet das Coisas, entre eles, Relés, Sensores, Servo Motores. O Raspberry PI 3 possui suporte a redes sem fio sem necessitar de adaptadores. Há Wi-Fi nos padrões B, G e N (802.11n) e Bluetooth 4.1 LE (Low Energy), e ainda possui a entrada de 40 pinos GPIO, 4 portas USB 2.0, slot para cartão micro SD (a versão SD foi utilizada somente na primeira versão, uma decisão feita para economizar espaço e considerando a maior popularidade dos cartões microSD), interface de câmera (CSI) e de display (DSI) e porta ethernet.

2.2.4.3. Galileo

O processador escolhido para equipar a placa é o Quark SoC X1000, um single-core muito parecido com um Pentium com clock de 400 MHz. A Galileo combina a performance da Intel com a expertise da Arduino na área open source, permitindo que ele rode Linux e uma vasta biblioteca de softwares livres. A nova placa terá suporte a desenvolvimento via Windows, Linux e Mac OS X e será compatível com diversas tecnologias como USB 2.0, PCI Express e Ethernet. Além do Processador Intel® Quark SoC X1000 de 400 Mhz (32 bits – núcleo único) ela contém 512 KB de SRAM, Memória DRAM de 256 MB, DDR3, Bootloader de 8 MB, Porta Ethernet 10/100 e é compatível com cartões SD de até 32 GB.

3. Message Queue Telemetry Transport (MQTT)

MQTT é um protocolo que visa prover conectividade para ambientes IoT e comunicação de máquina para máquina, tradução livre do conceito machine-to-machine(M2M). Esse protocolo foi projetado para ser extremamente leve, operar com baixo consumo de energia e pacotes de dados minimizados, além de realizar uma eficiente distribuição de informação para um ou mais receptores, ideal para aplicações móveis e situações onde o consumo de energia, link de comunicação com baixa capacidade de transmissão e recursos de processamento limitados são pontos críticos onde se deseja operar [MQTT.ORG,2016].

Esse protocolo de mensageria foi concebido para trabalhar baseado no princípio publish/subscribe para o transporte de mensagens(figura 4). Esse conceito consiste em vários clientes se conectam a um elemento centralizador, denominado broker, e para receber mensagens de determinadas áreas subscrevem tópicos, divisões hierárquicas dos assuntos de forma a organizar e identificar o assunto das mensagens publicadas, sendo esse o princípio responsável pela otimização da entrega de mensagens de um para um ou um para muitos. O broker funciona como uma interface que conecta a todos da rede, seu modo de operação facilita a entrada de novos sensores na rede sem a necessidade de criar um tópico, tendo em vista que o broker organiza-os por hierarquia usando uma barra como separador, algo semelhante a organização de pastas nos sistemas operacionais disponíveis no mercado, por exemplo, se em uma rede de sensores de temperatura, divididos em áreas de uma casa podem publicar as temperaturas mensuradas no seguinte tópico: *sensores/area1/temperatura/sensor1*, e qualquer outro dispositivo que desejar obter essas informações, é necessário que o cliente subscreva o tópico[MOSQUITTO.ORG,2016].

Dentre os motivos responsáveis pela escolha desse protocolo, podemos destacar a facilidade que ele oferece para implantação, ser um protocolo leve capaz de prover um consumo eficiente de energia e largura de banda, característica essencial para atender a IoT.

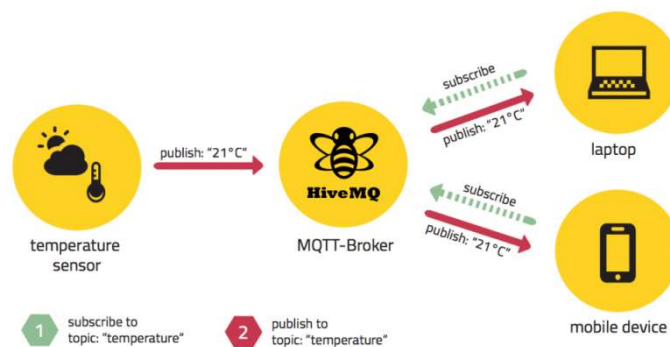


Figura 4 - Representação funcionamento básico MQTT, princípio publish/subscribe
(HIVEMQ.COM,2016)

3.1. Formato da mensagem MQTT

O protocolo MQTT trabalha com a troca de mensagens por um caminho bem determinado. Através de mensagens de controle formadas por até 3 partes. Cada mensagem do MQTT será formada por um cabeçalho fixo, presente em todas as mensagens, pode apresentar, também, uma segunda parte do cabeçalho de tamanho variável e o *payload*, que igualmente a parte variável, não se faz necessário em todas as mensagens, sendo este último a carga de dados a ser transmitida. Abaixo iremos descrever o formato para cada parte que compõem a mensagem, observando que as especificações que serão citadas são referentes ao MQTT V3.1[<http://public.dhe.ibm.com/>].

3.1.1. Cabeçalho Fixo

O cabeçalho fixo é a parte que está presente em cada mensagem de comando do MQTT. Na tabela 1, temos a representação dos campos, Message Type, DUP flag, QoS Level, RETAIN e suas respectivas posições.

bit	7	6	5	4	3	2	1	0
byte 1	Message Type				DUP flag	QoS level		RETAIN
byte 2	Remaining Length							

Tabela 2 - Cabeçalho fixo mensagem MQTT.

Message Type: Corresponde aos bits 7-4 do primeiro byte do cabeçalho. Com uma representação de 4 bit sem sinal, em uma próxima seção serão listados os possíveis tipos de mensagem do MQTT.

DUP flag: Posicionada no bit 3 do primeiro byte, esta flag só é acionada quando o cliente ou o servidor tenta reenviar uma mensagem do tipo PUBLISH, PUBREL, SUBSCRIBE ou UNSUBSCRIBE. Vale ressaltar que a DUP flag é utilizada, somente, quando o valor do QoS é maior que zero, e quando confirmações forem necessárias.

QoS Level: No MQTT são previstos 3 níveis de Qualidade de Serviço, determinados pelo valor que se encontra no primeiro byte do cabeçalho, bits 2-1. O QoS irá definir o quão o cliente/broker irá se empenhar para garantir que uma mensagem seja publicada ou entregue à outro cliente. Níveis de QoS implementados no MQTT:

- ☐ QoS nível 0 – O broker/cliente irá entregar a mensagem uma vez, sem confirmar o recebimento da mensagem.
- ☐ QoS nível 1 – O broker/cliente irá entregar a mensagem pelo menos uma vez, realizando a confirmação necessária para garantir a entrega da mensagem.
- ☐ QoS nível 2 – O broker/cliente irá entregar a mensagem somente uma vez utilizando por meio de um *handshake* de 4 etapas.

Se observar os níveis de QoS implantados no MQTT, mas devemos nos atentar que quanto maior o nível de qualidade exigido, maior será a confiança na troca de mensagens mas essa escolha implicará maior latência na comunicação e maior largura de banda será um requisito desta comunicação[MOSQUITTO.ORG,2016].

RETAIN: Posicionado no bit 0 do primeiro byte, esta flag é utilizada somente nas mensagens do tipo PUBLISH. Quando o cliente envia uma mensagem ao servidor com a flag RETAIN ativada, indica que o servidor deve reter a mensagem, mesmo que todos os atuais assinantes já tenham recebido a mensagem publicada. Quando uma nova assinatura chega ao servidor, a última mensagem retida deve ser enviada com a flag sinalizada.

3.1.2. Tipos de mensagens MQTT

Na tabela 3 estão representadas os tipos de mensagens que podem ser enviadas pelo protocolo MQTT.

Mnemônico	Código	Descrição
Reserved	0	Reservado
CONNECT	1	Requisição do cliente para conectar com servidor
CONNACK	2	Confirmação de conexão
PUBLISH	3	Publicar mensagem
PUBACK	4	Confirmação de publicação
PUBREC	5	Mensagens de confirmação de entrega com QoS
PUBREL	6	Mensagens de confirmação de entrega com QoS
PUBCOMP	7	Mensagens de confirmação de entrega com QoS
SUBSCRIBE	8	Cliente requisita assinatura
SUBACK	9	Confirmação de assinatura
UNSUBSCRIBE	10	Cliente requisita encerramento da assinatura
UNSUBACK	11	Confirmação de encerramento da assinatura
PINGREQ	12	PING Request
PINGRESP	13	PING Response
DISCONNECT	14	Cliente desconectando
Reserved	15	Reservado

Tabela 3 - Tipos de mensagens MQTT

CONNECT: Esse tipo de mensagem é enviada pelo cliente após estabelecer conexão com o servidor via socket TCP/IP, criando uma sessão do MQTT. Na mensagem CONNECT, as flags DUP, QoS e RETAIN não são utilizadas, e no segundo byte temos o tamanho do cabeçalho variável e o tamanho do *payload*.

CONNACK: Mensagem retornada ao cliente com resposta à solicitação de conexão. Assim como a mensagem CONNECT as flags do cabeçalho fixo não são utilizadas, e no cabeçalho variável se encontra o código de retorno, podendo assumir qualquer um dos valores listados na tabela 4 e não possui payload.

Enumeration	HEX	Meaning
0	0x00	Connection Accepted
1	0x01	Connection Refused: unacceptable protocol version
2	0x02	Connection Refused: identifier rejected
3	0x03	Connection Refused: server unavailable
4	0x04	Connection Refused: bad user name or password
5	0x05	Connection Refused: not authorized
6-255		Reserved for future use

Tabela 4 - Respostas para mensagem CONNECT

PUBLISH: Mensagem enviada pelo cliente para o broker, para que este redistribua a mensagem aos interessados subscritores, cada mensagem é associada a um tópico. Os tópicos são organizados por nomes e organiza as informações de forma hierárquica. Uma mensagem publicada em um determinado tópico deve ser entregue à todos aqueles que realizaram a assinatura daquele tópico.

PUBACK: Uma mensagem PUBACK é enviada de um servidor para cliente, em resposta a uma mensagem do tipo PUBLISH, ou de um cliente para servidor ao receber uma mensagem do tipo PUBLISH. Devemos atentar para o fato que a mensagem PUBACK acontecerá, somente, quando o QoS da mensagem de publicação estiver setado como QoS level 1.

Como explicado anteriormente, no QoS de nível 2, a mensagem é entregue seguindo um fluxo do MQTT de 4 etapas. Sendo as próximas 3 mensagens, participantes desse fluxo.

PUBREC: Mensagem enviado pelo servidor a um cliente ou de um cliente subscritor para um servidor em resposta a uma mensagem do tipo PUBLISH, sendo também a segunda mensagem do fluxo de controle de entrega do MQTT para o QoS nível 2.

PUBREL: Terceira mensagem que visa cumprir a entrega garantida em 4 etapas. É enviada em resposta a mensagem PUBREC, seja ela enviada pelo servidor ou por um cliente.

PUBCOMP: Finalizando o *handshake* de 4 etapas, PUBCOMP é a mensagem que finaliza o fluxo do QoS nível 2 e indica que a mensagem foi entregue com sucesso.

SUBSCRIBE: Essa mensagem permite que um cliente se registre em tópicos de seu interesse. Mensagens publica nesses tópicos serão entregues aos subscritores pelo servidor. A parte do pacote que merece destaque são os campos que indicam o tópico que o cliente deseja assinar e o nível do QoS com que as mensagens devem ser entregues.

UNSUBSCRIBE: Mensagem enviada de um cliente para um servidor para retirar inscrição em um tópico.

SUBACK: Mensagem que sai do servidor para um cliente que solicitou a inscrição em determinado tópico. Confirmando ou não sua subscrição.

UNSUBACK: O servidor responde uma mensagem UNSUBSCRIBE com uma UNSUBACK confirmando que recebeu o pedido para retirar a inscrição do cliente e com um identificador de pacotes para confirmar a retirada da inscrição.

PINGREQ: A mensagem PINGREQ é enviada do cliente para o servidor, e é usada para:

1. Indicar ao servidor que o cliente está ativo quando ele não tem nenhum outro pacote para enviar ao servidor
2. Para o cliente confirmar que o servidor está ativo
3. Testar se a rede está ativa

PINGRESP: Mensagem enviada do servidor para o cliente em resposta a um PINGREQ, indica que o servidor está ativo.

DISCONNECT: É a mensagem final do cliente a um servidor, indica que o cliente encerrou a conexão que havia entre ambos. Depois de mandar um pacote DISCONNECT o cliente fecha a conexão com a rede e não envia mais nenhum pacote de controle e se o cliente não fechar a conexão o servidor deverá fechar.

3.2. Fluxos MQTT e Níveis de Serviço (QoS)

O fluxo utilizado pelo MQTT para entregar mensagens vai depender do nível de qualidade de serviço solicitado para a entrega. Abaixo temos diferentes formas que podem ser utilizadas para implementar o funcionamento do protocolo.

Fluxo de mensagem com QoS nível 0: Na figura 5 temos a representação deste caso, aparece a forma mais simples de entrega de mensagem. Não existe a troca de mensagens de controle para confirmar a entrega do pacote. O emissor enviará a mensagem e, caso ocorra tudo dentro dos conformes, a mensagem chegará ao receptor. Sem mensagens de confirmação ou garantia de entrega.

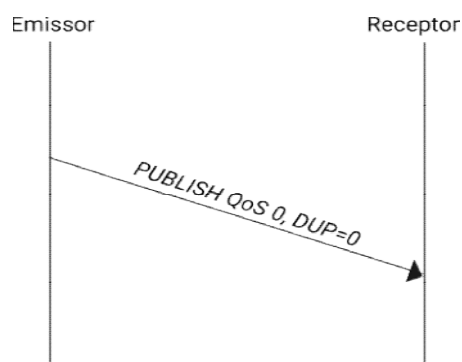


Figura 5 - Fluxo de mensagens MQTT com QoS nível 0

Fluxo de mensagem com QoS nível 1: Neste nível de serviço, o objetivo é a garantia de que a mensagem será entregue, pelo menos, uma vez. O pacote PUBLISH com o QoS nível 1 terá um campo no cabeçalho variável com um identificador que ainda não foi utilizado marcando o pacote, sendo esse o valor utilizado na confirmação do pacote PUBACK retornado pelo receptor. O fluxo da troca de mensagens é ilustrado na figura 6.

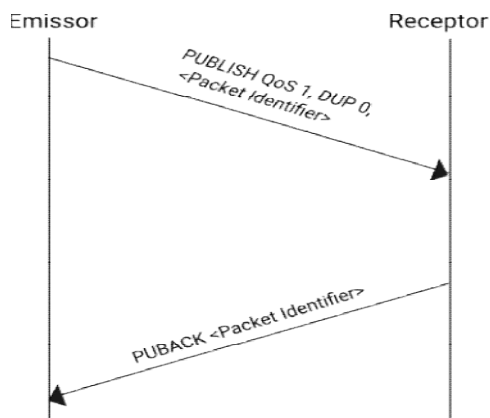


Figura 6 - Fluxo de mensagens MQTT com QoS nível 1

Fluxo de mensagem com QoS nível 2: É o mais elevado nível de QoS implementado para o MQTT, quando a aplicação não permite nenhuma perda ou duplicação de pacote este nível deverá ser utilizado. Neste caso, a confirmação é realizada em 4 etapas. O fluxo apresentado na figura 7 se inicia quando o emissor marca uma mensagem do tipo PUBLISH, na intenção de transmitir dados para outros dispositivos ou aplicações, com um identificador único e envia. O receptor, em resposta a primeira mensagem retorna um pacote do tipo PUBREC, com o mesmo identificador da mensagem que iniciou o fluxo. Na 3ª etapa, o emissor enviará um pacote PUBREL, que será considerado pacote sem confirmação até que ele receba uma mensagem de confirmação, PUBCOMP, do servidor, indicando que a transmissão foi bem sucedida.

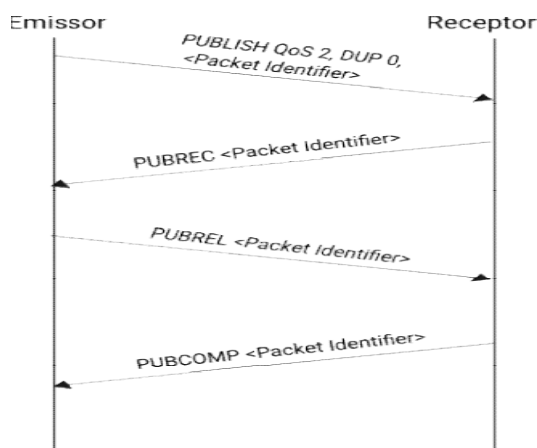


Figura 7 - Fluxo de mensagens MQTT com QoS nível 2

3.3.3. Segurança do protocolo MQTT

A segurança implementada para o MQTT está baseada nos três pilares da segurança, identificação, autenticação e autorização. Para manter o ambiente seguro ele trabalha com a Identificação, criando identidades únicas para cada Cliente MQTT, no segundo passo vem a autenticação, momento de verificar a autenticidade da identidade apresentada. Em seguida temos o terceiro pilar, a autorização que consiste em verificar se mesmo com credenciais válidas o cliente MQTT tem autorização para executar o comando x ou acessar a informação y.

Assim como nos ambientes IoT, a parte de segurança implementada para o MQTT se divide em camadas. Onde cada uma é voltada para uma diferente abordagem de como o sistema pode ser atacado. Na camada de rede é possível utilizar VPN para proteger a conexão entre os dispositivos e servidores. Na camada de transporte, são utilizadas técnicas de criptografia para cifrar a mensagem

evitando que o sigilo de dados do sistema seja comprometido. Usualmente são utilizadas os métodos SSL e TLS. Na camada de aplicação temos o MQTT, fornecendo a capacidade de identificação dos usuários e credenciais com usuário e senha como implementação do próprio protocolo. Processos como autorização ou permissões ficam a cargo da implementação do broker.

4. PROPOSTA DE ALTERAÇÃO DO PROTOCOLO MQTT

Neste capítulo se apresenta a proposta de alteração do protocolo MQTT para atingir o objetivo do projeto de fornecer um protocolo que promova o controle de acesso às informações na camada de aplicação.

4.1. Ambiente de simulação

Para realizar essa proposta foi utilizada a topologia citada abaixo, onde se tem os principais personagens dessa troca de mensagens: Máquina 1, Broker MQTT e uma aplicação, representados na figura 8.

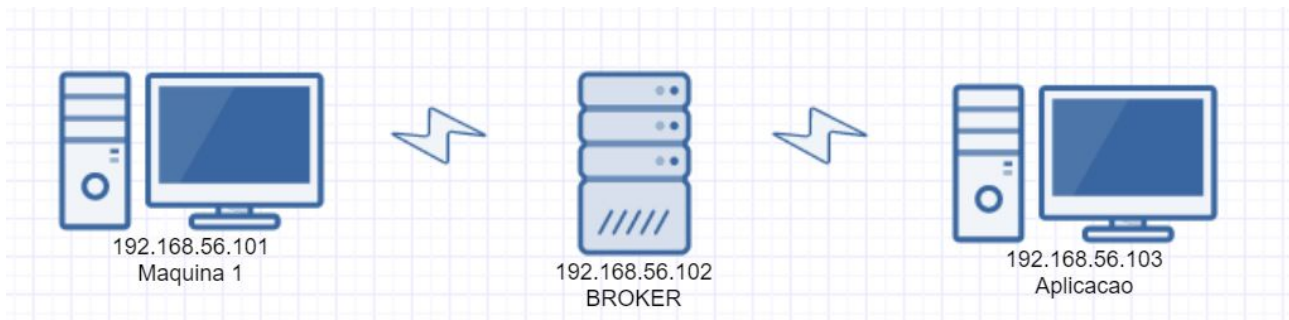


Figura 8 - Topologia utilizada na simulação

Máquina 1: Quando se fala de IoT logicamente, se imagina em sensores promovendo a entrada dos dados na rede, entretanto para efeitos de simulação iremos fazer uso da Máquina 1, como se ela fosse o responsável por um serviço crítico e tivéssemos a necessidade de verificar o processamento da máquina durante todo o tempo. Trata-se de uma máquina Linux, onde foi implementado um script que captura a informação desejada e publica como mensagem MQTT no tópico “*maquinas/1/cpu*” a cada 15 segundos utilizando a ferramenta *crontab*.

```
#!/bin/bash
#Script executado a cada 15s gerando dados sobre o processamento da máquina

ip_address_broker="192.168.56.102" #Endereço IP do Broker
topic_name="maquinas/1/cpu" #Tópico onde iremos publicar as mensagens

#mpstat e awk para capturar os dados: Data, hora e processamento
mpstat | awk 'BEGIN {ORS="\t"} NR ==1 { print $4 } NR ==4 {print $1"\t"$3};'
> get_stream.txt

echo " " >> get_stream.txt #insere newLine

message=$(awk 'NR==1 {print $0}' get_stream.txt) #recupera dados do arquivo

#mensagem para publicar a informação
mosquitto_pub -d -h $ip_address_broker -t $topic_name -m "$message"
```

Broker MQTT: Para montar o Broker foi utilizado um servidor com a distribuição Ubuntu Server e o Mosquitto como software que implementa o Broker MQTT na versão 3.1 do protocolo, pois além de ser software livre é de fácil instalação e utilização.

Aplicação: A terceira máquina tem a função de simular uma aplicação consumindo, ou melhor dizendo, subscrevendo os tópicos do broker, para receber as informações de processamento da máquina1. Temos representado na figura abaixo.

Com essa topologia fizemos algumas simulações com a ajuda de três máquinas virtuais, cada uma fazendo uma função específica. Fizemos simulações alterando as qualidades de serviço de 1 a 3. Usando o wireshark para capturar pacotes pela rede, foi capturado primeiramente a conexão e a publicação entre a máquina 1 e o broker usando um QoS 0 (Sem controle de qualidade). Os dados capturados são representados na figura baixo.

14	39.835309	192.168.56.101	192.168.56.102	MQTT	105	Connect Command
16	39.835486	192.168.56.102	192.168.56.101	MQTT	70	Connect Ack
18	39.836687	192.168.56.101	192.168.56.102	MQTT	91	Publish Message
19	39.837458	192.168.56.101	192.168.56.102	MQTT	68	Disconnect Req

Figura 9 - Captura wireshark (conexão entre maquina e broker)

Em seguida capturamos os pacotes que mostram a conexão entre a aplicação e o broker e também a aplicação subscrevendo o tópico para receber as informações. A troca de mensagens está representada na figura10.

4	0.001420	192.168.56.103	192.168.56.102	MQTT	105	Connect Command
6	0.001562	192.168.56.102	192.168.56.103	MQTT	70	Connect Ack
8	0.002870	192.168.56.103	192.168.56.102	MQTT	83	Subscribe Request
9	0.002946	192.168.56.102	192.168.56.103	MQTT	71	Subscribe Ack

Figura 10 - Captura wireshark(conexão entre aplicação e broker)

E por ultimo, na figura 11 temos o Broker enviando a mensagem que a maquina 1 publicou para a aplicação.

20	39.837567	192.168.56.102	192.168.56.103	MQTT	91	Publish Message
----	-----------	----------------	----------------	------	----	-----------------

Figura 11 - Captura wireshark (broker enviando mensagem para aplicação)

Logo em seguida fizemos teste com QoS 1, na figura 12 pode-se notar um maior controle de qualidade, logo depois que a maquina publica no broker, este envia uma confirmação dessa mensagem recebida (mensagem 19 em destaque)

14	5.367499	192.168.56.101	192.168.56.102	MQTT	105	Connect Command
16	5.367700	192.168.56.102	192.168.56.101	MQTT	70	Connect Ack
18	5.372494	192.168.56.101	192.168.56.102	MQTT	93	Publish Message
19	5.372620	192.168.56.102	192.168.56.101	MQTT	70	Publish Ack

Figura 12 - Captura wireshark (conexão entre maquina e broker com QoS 1)

Por ultimo foi colocado o QoS em 2, aonde tem um controle de entrega muito maior, como se pode ver na figura a seguir onde ocorre um four way handshake. Na figura 13 é possível visualizar todas as mensagens de confirmação utilizadas para garantir a entrega.

14	14.021894	192.168.56.101	192.168.56.102	MQTT	105	Connect Command
16	14.022068	192.168.56.102	192.168.56.101	MQTT	70	Connect Ack
18	14.024207	192.168.56.101	192.168.56.102	MQTT	93	Publish Message
19	14.024306	192.168.56.102	192.168.56.101	MQTT	70	Publish Received
20	14.025625	192.168.56.101	192.168.56.102	MQTT	70	Publish Release
21	14.025713	192.168.56.102	192.168.56.101	MQTT	70	Publish Complete

Figura 13 - Captura wireshark (conexão entre maquina e broker com QoS 2)

Depois disso, pautou se as alterações, tentando utilizar os espaços existentes no MQTT, realizar o menor número de alterações possíveis, visando manter o maior grau de compatibilidade com as versões anteriores. Seguimos, também, a idéia de incrementar o quesito segurança da versão atual do protocolo, que atualmente se encontra na versão 3.1. Atualmente, as opções de segurança oferecidas são, a capacidade de criptografar os dados enviados pela rede, onde os algoritmos mais utilizados são, SSL e TLS. O MQTT, também fornece um modo de autenticação por usuário e senha. As demais configurações, de acesso e autorização ficam a cargo da configuração do broker. O objetivo do nosso trabalho é fazer com que nas especificações do protocolo, além dos itens já implementados no funcionamento deste padrão, seja realizado o controle do nível de acesso, como atividade inerente ao funcionamento do MQTT, normalizando a funcionalidade e aumentando o grau de segurança do protocolo de comunicação projetado pela IBM.

4.2. Controle de Nível de Acesso

Nesta seção será demonstrado os requisitos que devem ser implementados para promover o controle e gerenciamento dos níveis de acesso e como os dispositivos.

4.2.1. Estabelecendo os limites dos tópicos

4.2.1.1. Classificação dos tópicos

Cada tópico pode ser marcado para poder ser acessado por qualquer um que tenha a intenção de publicar ou subscrever, ou então restringir o acesso apenas aos que possuem autorização para a operação atribuindo os valores *private* ou *public* para o tópico.

4.2.1.2. Permissões dentro dos tópicos

Outra maneira de realizar o controle é estabelecendo quem poderá interagir com o tópico. Dessa forma podemos registrar quais grupos ou quais usuários poderão acessar ou modificar as informações que passam pelo tópico.

4.2.2. Credenciais de acesso

4.2.2.1. Credenciais para usuários

Quando se passa para o ponto de vista do dispositivo, podemos determinar quais são os tópicos que o dispositivo pode acessar e de que forma. Sendo assim o gerente da rede irá dizer qual o tópico e se o usuário poderá publicar, subscrever ou realizar ambas as funções.

4.2.2.2. Segmentação de Domínios

Para facilitar o controle de credenciais, podemos optar por dividir os dispositivos em grupos, marcando-os com as tags dos grupos dos quais eles participam. Dessa forma ele terá acesso somente aos tópicos que estão destinados aqueles grupos.

4.2.2.3. Tempo de guarda

Pode ser determinado, também, um intervalo de tempo em que o dispositivo deve respeitar no envio entre uma mensagem e outra para o broker.

4.3. Função de Registro

O primeiro passo é estabelecer um padrão como parâmetro para análises futuras, pois a função de registro foi concebida, para utilizar o conceito da IoT de análise do contexto e ser capaz de alterar a resposta adequada para cada situação que seja apresentada ao nosso dispositivo. A mensagem CONNECT é utilizada para que o dispositivo informe ao broker sua identidade e realize a autenticação para poder subscrever ou publicar em tópicos ativos na rede. A nossa proposta diz que a partir de agora o dispositivo irá, também, informar no momento da sua primeira conversa com o broker, o tópico que deseja acessar, e a operação a ser realizada. Entregando a trinca: Quem sou, O que vou fazer, Aonde vou fazer. Na segunda etapa o broker terá o trabalho de validar a

solicitação, atribuir um peso ao registro e atribuir uma data de validade as credenciais do dispositivo ponderada pelo peso calculado na etapa anterior, o broker deverá realizar as seguintes verificações:

Dispositivo já possui registro? A ideia deste procedimento é verificar se em algum momento o elemento já foi registrado. Em primeiro plano, o procedimento visa que um mesmo dispositivo registre-se mais de uma vez e também que outros dispositivos tentem realizar o registro e fazer uso das credenciais de outro elemento da rede.

O dispositivo já foi registrado? Existe a possibilidade do registro existir, porém a data de validade já foi alcançada, gerando a obrigação de uma nova fase de registro para o dispositivo, e a tarefa de verificar a relação entre o novo registro e as confirmações anteriores a fim de identificar eventuais divergências, visando eliminar a possibilidade de que equipamentos possam publicar ou subscrever tópicos alheios ao seu campo de atuação.

4.3.1. Formato mensagem CONNECT

É importante ressaltar que devemos implementar nossas ideias sem alterar o formato já utilizado na versão mais atual do protocolo MQTT. A primeira mudança proposta é acrescentar o campo *Register Flag*, para sinalizar quando o dispositivo deseja realizar o seu registro e poder utilizar os serviços daquele Broker. Na tabela abaixo temos representado o formato da mensagem CONNECT, onde o dispositivo cliente informa ao broker os parâmetros da sua conexão e a presença ou ausência de campos na carga útil do pacote.

Está em destaque o bit 0 do décimo byte de dados, responsável por indicar os valores das flags utilizadas pelo MQTT. O Bit em questão é reservado para validação do campo de flags, de acordo com as especificações o servidor deverá desconectar o cliente, caso seja atribuído um valor diferente de 0 para o bit menos significativo deste campo. Essa decisão não contribui em nenhum aspecto para o funcionamento do protocolo, pois não é uma validação dos campos anteriores, sendo assim o único caso em que ele iria ajudar seria se o próprio bit fosse alterado na transmissão. Levando em consideração que o tratamento de troca de bits pode ser realizado de maneira mais eficaz em outras camadas, essa validação torna-se irrelevante para o processo. Dessa forma decidimos utilizá-lo para sinalizar a função Register flag, sem incrementar a quantidade de dados na transmissão. Quando Register flag for igual a 1 o broker deverá executar a função de Registro e realizar os procedimentos já explicados no tópico anterior. Quando Register flag igual a 0(zero), o broker precisará, somente, verificar se o dispositivo está devidamente registrado. Em caso afirmativo, deverá continuar com o processo de conexão que já era feito anteriormente, para validar ou não a tentativa de conexão do dispositivo.

Bit	7	6	5	4	3	2	1	0	Description
Fixed Header	Message Type (1)				DUP flag	QoS level		RETAIN	
byte 1	0	0	0	1	X	X	X	X	
byte 2	Remaining Length								
Protocol Name									
Byte 3	0	0	0	0	0	0	0	0	Length MSB (0)
Byte 4	0	0	0	0	0	1	0	0	Length LSB (4)
Byte 5	0	1	0	0	1	1	0	1	‘M’
Byte 6	0	1	0	1	0	0	0	1	‘Q’
Byte 7	0	1	0	1	0	1	0	0	‘T’
Byte 8	0	1	0	1	0	1	0	0	‘T’
Byte 9	0	0	0	0	0	1	0	0	Level(4)
byte 10	1	1	0	0	1	1	1	X	User name flag (1)
									Password flag (1)
									Will RETAIN (0)
									Will QoS (01)
									Will flag (1)
Clean Session (1)									
Keep Alive timer									
byte 11	0	0	0	0	0	0	0	0	Keep Alive MSB (0)
byte 12	0	0	0	0	1	0	1	0	Keep Alive LSB (10)

Tabela 5 - Formato mensagem MQTT

4.4. Broker

O Broker será a peça fundamental nesse processo, pois será o responsável pelo controle do envio das mensagens, por consequência será o responsável pelas verificações de tipo de acesso e outras variáveis.

4.4.1. Recebendo solicitações de conexão

Anteriormente, foi discutida uma nova forma de tentar conectar-se ao broker pela primeira vez, falando da função Registro. Para o funcionamento correto da nossa proposta precisa se estabelecer algumas premissas sobre o ambiente em que o nosso sistema de mensageria está inserido. Devemos levar em consideração que existe um módulo realizando o controle dos dispositivos de uma rede IoT e o gerenciamento dos níveis de acesso das informações.

Assim sendo, o trabalho do broker irá mudar de modo que não irá, somente, controlar o fluxo das mensagens, mas verificará quando o dispositivo utilizar a mensagem CONNECT com a intenção de realizar o seu registro, ou seja, a flag para função registro está setada como 1, conexões posteriores seguirão o fluxo normal já existente.

4.4.1.1. Função Registro – Visão Broker

Após receber uma mensagem do tipo CONNECT, o broker deve primeiramente fazer as leituras das *flags*, e proceder para cada uma como já vem determinado nas especificações do protocolo. A mudança significativa será na leitura do bit modificado, representado a flag *register*, quando sinalizado com o valor 1, a mensagem indicará que o elemento quer realizar o registro da sua identidade e características para se conectar ao broker e poder trocar mensagens. Ao identificar que é uma mensagem que tem a intenção de registrar um dispositivo na rede, o broker irá coletar os dados informados na mensagem, que deverá conter a informação de registro nos moldes da estrutura apresentada a seguir. O segundo passo será verificar se as informações fornecidas pelo dispositivo, estão de acordo com o as permissões liberadas pelo administrador da rede. Após a verificação dos dados e validação pelo Broker, será retornada a mensagem CONNACK que carrega no cabeçalho variável o código de resposta a solicitação de conexão. Como foi detalhado em seção anterior, onde foi explicado o funcionamento do MQTT, sabemos que para essa função são especificados cinco códigos de retorno para tentativa de conexão, no caso em que as permissões solicitadas pelo dispositivo forem divergentes das permissões concedidas pelo administrador da rede será retornado o código 0x05 que corresponde a mensagem: *Connection Refused: not authorized*, para os outros casos se continuou a seguir as especificações da atual versão do protocolo. Para finalizar este tópico temos a seguir a demonstração gráfica do novo fluxo de conexão criado pela proposta do trabalho, e também a estrutura de organização das informações no dispositivo e no broker.

4.4.1.2. Novo fluxo de conexão MQTT

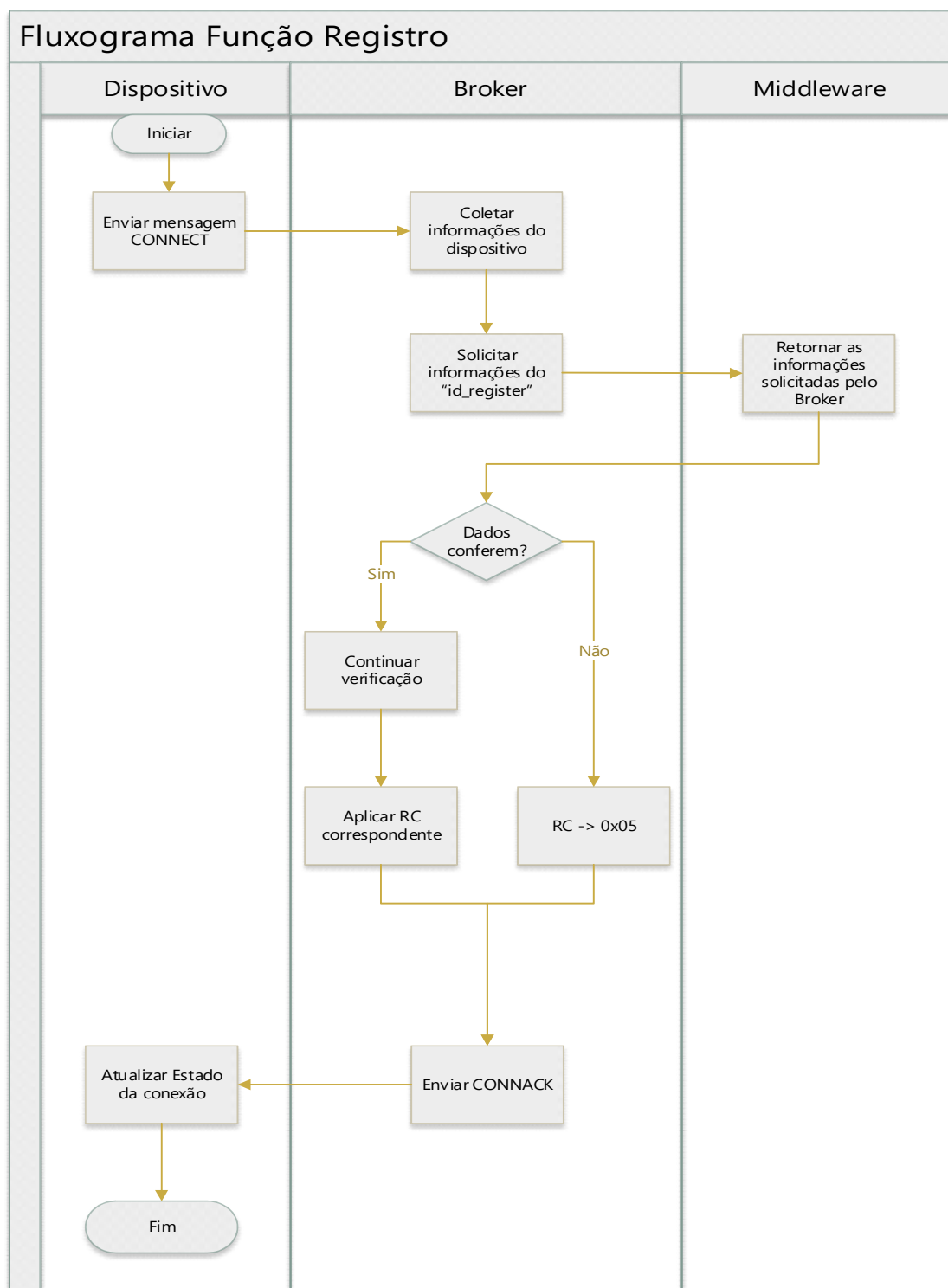


Figura 14 - Fluxograma da função registro

4.4.1.3.Estrutura do Registro:

```
<Registro id_register="id" username="user">
  <topics>
    <topic topic="name" action="read"/>
    <topic topic="name" action="read"/>
    <topic topic="name" action="write"/>
    ...
    <topic topic="name" action="read/write"/>
  </topics>
</Registro>
```

Campo	Descrição
id_register	Identificador único do dispositivo. Comumente identificado pelo macAddress.
Username	Nome escolhido para facilitar na dos dispositivos pelos usuários.
Topic	Tópico de interesse
Action	Ação que será realizada pelo dispositivo.

Tabela 6 - Descrição das informações repassadas pelo dispositivo na função registro

4.4.2. Análise do Contexto

Considerando as limitações presentes nos dispositivos menores, devemos optar por colocar qualquer carga de processamento no broker que, naturalmente, deve ser uma máquina mais robusta por representar o ponto onde todas as mensagens são centralizadas e distribuídas para os seus respectivos assinantes. Por isso, entendemos que as atividades de controle deverão ser organizadas e tratadas pelo broker.

4.4.2.1. Analisando mensagens em relação ao tipo de acesso

Foi colocada como a primeira verificação, a observação da mensagem enviada pelo dispositivo e se ela está de acordo com o tipo de acesso verificado no momento do registro do dispositivo. No momento em que houver uma solicitação de subscrição de determinado tópico, o broker irá verificar se aquele dispositivo específico tem autorização para ler e se tem autorização para ler aquele tópico específico. Da mesma forma acontecerá para leitura. Dessa forma, evitaremos que dispositivos estranhos consigam acessar dados do sistema ou fornecer dados equivocados com a intenção de corromper a confiabilidade desses dados. Esse tipo de verificação servirá para defender, principalmente, os nós das bordas do sistema, pois, geralmente, são eles os responsáveis por estabelecer conexões ou serem atuadores nos ambientes em que foram instalados. Sendo assim conseguimos reduzir o campo de atuação dos invasores em ataques como do tipo Hello flood, em que dispositivos tentam se passar por nós da rede. No caso de conseguirem a clonagem, os invasores não terão acesso aos outros tópicos existentes.

4.4.2.2. Analisando mensagens em relação ao tempo

O broker irá analisar o envio de mensagens em relação à variável tempo, onde pode ser indicado alguns parâmetros para determinar se o comportamento de dispositivo é prejudicial ou não ao funcionamento da aplicação.

Tempo de intervalo entre mensagens: O broker poderá negociar com elemento que está se registrando na rede, um intervalo de tempo mínimo que terá q ser respeitado entre o envio de uma mensagem e outra. Desse forma, sempre que um dispositivo começar a sobrecarregar o broker com o envio de mensagens fora do tempo acordado, a conexão deverá ser cortada. Fornecendo um mecanismo de defesa na camada de aplicação, contra ataque DoS.

Tempo de inatividade: Nesse caso, os dispositivos serão classificados em ordem por tempo de inatividade. O tempo de corte será definido pelo gestor da rede, e todos que ultrapassarem esse limite terão suas autorizações revogadas e para voltar terão que realizar a função registro novamente.

5. Conclusão

Nesta dissertação, se apresentou uma proposta de modificação do protocolo MQTT, capaz de fornecer mecanismos de defesa contra alguns ataques mais comuns em redes IoT, sem comprometer ou onerar o funcionamento dos dispositivos, tendo em vista que foi aproveitada a arquitetura utilizada pelo protocolo e foi concentrada a carga de processamento no Broker, que é um computador com maior poder computacional. Trabalhando dessa forma será possível aumentar a capacidade do protocolo de trabalhar de uma maneira segura respeitando as limitações impostas pelas características do ambiente IoT, dentre elas podemos citar a capacidade de processamento reduzida e a preocupação com a eficiência energética dos dispositivos.

A pesquisa bibliográfica realizada identifica os principais aspectos da segurança envolvendo a IoT, os quais foram fundamentais e imprescindíveis para o desenvolvimento do trabalho e definição de quais alterações deveriam ser propostas para a arquitetura final.

A proposta de alteração apresentada neste trabalho mostrou-se promissora pois consegue deixar o ambiente mais seguro atendendo as necessidades da Internet das Coisas, entretanto não temos a certeza que será utilizada, pois através das pesquisas percebemos que ainda não foi estabelecido um padrão e dentre os vários protocolos existentes e que fazem concorrência com o MQTT, todos têm potencial para serem escolhidos.

5.1. Trabalhos Futuros

Como proposta de trabalhos futuros são indicados alguns pontos que podem ser usados para dar continuidade ao trabalho.

Implementação da proposta deste trabalho. Alterando o código do protocolo MQTT, para implementar as funções dos dispositivos, bem como as alterações na rotina do Broker.

Existe a necessidade de realizar um estudo mais aprofundado sobre os diferentes nichos de atuação da IoT, definir cenários e como o aspecto de segurança deve ser enfrentado em cada situação.

6. REFERÊNCIAS BIBLIOGRÁFICAS

ALLIANCE, W.-F. (2015). Who we are.

ARAUJO, Assis. Algoritmos de Criptografia (DES, 3DES, AES, RC4) <<http://tecnologiadarede.webnode.com.br/news/noticia-aos-visitantes/> />. Acesso em: 22 de Nov. de 2016.

CARISSIMI, Alexandre. Internet das Coisas, middlewares e outras coisas,2016.

EVANS, Dave. The internet of things. How the Next Evolution of the Internet is Changing Everything, Whitepaper, Cisco Internet Business Solutions Group (IBSG), v. 1, p. 1-12, 2011.

FORBES <http://www.forbes.com/sites/gilpress/2014/08/22/internet-of-things-by-the-numbers-market-estimates-and-forecasts/#20192b532dc9>, acessado em 15/06/2016.

FRIAS, Renato N. ZigBee, 2004 Disponível em:

<http://www.teleco.com.br/tutoriais/tutorialzigbee/pagina_1.asp />. Acesso em 12 de nov. 2016.

HIVEMQ Enterprise MQTT Broker; MQTT Essentials Part 2: Publish & Subscribe <<http://www.hivemq.com/blog/mqtt-essentials-part2-publish-subscribe> /> Acesso em 28 de Nov. de 2016.

LACERDA, Flavia; LIMA-MARQUES, Mamede. Da necessidade de princípios de Arquitetura da Informação para a Internet das Coisas. Perspectivas em Ciência da Informação, v. 20, n. 2, p. 158-171, 2015.

LOCATO, Felipe. IoT: uma tendência, muitas vulnerabilidades <<https://www.binarionet.com.br/blog/IoT-uma-tendencia-muitas-vulnerabilidades/> /> Acesso em 23 de Nov. de 2016.

MACEDO, Henrique. Algoritmos de criptografia leve. <http://calhau.dca.fee.unicamp.br/wiki/images/8/84/Algoritmos_de_Criptografia_Leve_v1.pdf />

MATEUS, Geraldo Robson; LOUREIRO, Antonio Alfredo Ferreira. Introdução à computação móvel. 1998.

MAHAJAN, Perna; SACHDEVA, Abhishek. A study of Encryption algorithms AES, DES and RSA for security. Global Journal of Computer Science and Technology, v. 13, n. 15, 2013.

MQTT Version 3.1.1. Edited by Andrew Banks and Rahul Gupta. 29 October 2014. OASIS Standard. <http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html>. Latest version: <http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html>.

MQTT V3.1 Protocol Specification, International Business Machines Corporation (IBM) Eurotech <public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html /> Acesso em 30 de Nov. de 2016.

NETO, Oliveira et al. Síntese de requisitos de segurança para internet das coisas baseada em modelos em tempo de execução. 2015.

NETO, Zamith França. Aplicação da tecnologia RFID para a gestão acadêmica, de pessoal e operacional de patrimônio e biblioteca. 2008.

PIRES, Paulo F. et al. Plataformas para a Internet das Coisas. Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, 2015.

REITER, G. (2014). Wireless connectivity for the internet of things: One size does not fit all. page 11.

SANTOS, B. P., Silva, L. A., Celes, C. S., Borges, J. B., Neto, B. S. P., Vieira, M. A. M., & Vieira, L. F. M. Internet das Coisas: da Teoria à Prática.

TANEMBAUM, A. (2011). Computer Networks. Prentice Hall Professional Technical Reference, 5th edition.

The TLS Protocol. T. Dierks, C. Allen. <<http://www.ietf.org/rfc/rfc2246.txt> /> Acesso em 04 de Dez de 2016.

O que é o Código Eletrônico de Produto (EPC)?, <<https://www.gs1br.org/faq/o-que-e-epc> /> Acesso em 12 de Abril de 2017.

RODRIGUES, Fábio Figueiredo; KLEINSCHMIDT, João Henrique. Estudo de Aplicações da Internet das Coisas em um Ambiente Acadêmico. Revista ESPACIOS| Vol. 35 (Nº 13) Año 2014, 2014.

WEISER, Mark. The computer for the 21st century. Scientific american, v. 265, n. 3, p. 94-104, 1991.

Redes locais sem fio que atendem ao padrão IEEE 802.11, Luis Guilherme Uzeda Garcia, <https://www.gta.ufrj.br/grad/01_2/802-mac/ > , Acesso em 21 de Abril de 2017.

Introdução a Mensageria, Concrete Solutions, 2012<
<https://www.concretesolutions.com.br/2012/01/23/introducao-a-mensageria/> > , Acesso em 26/04/2017.

Introdução ao AMQP com RabbitMQ < <http://www.devmedia.com.br/introducao-ao-amqp-com-rabbitmq/33036> > , Acesso em 26/04/2017.

Redes Sem Fio – Aspectos de Segurança < <http://www.ufrgs.br/tri/Documentos/redes-sem-fio-aspectos-de-seguranca> > , Acesso em 03/05/2017.

[1]<https://www.bluetooth.com/what-is-bluetooth-technology/bluetooth-fact-or-fiction> , acessado em 15/06/2016

[2]<https://www.opensensors.io/connectivity> , acessado em 15/06/2016

[3]<http://www.semtech.com/wireless-rf/lora/LoRa-FAQs.pdf> , acessado em 15/06/2016

[4] <https://www.lora-alliance.org> , acessado em 15/06/2016

[5] <http://makers.sigfox.com> , acessado em 15/06/2016

<http://mqtt.org/>, acessado em 02/07/2016

<http://mosquitto.org/man/mqtt-7.html> acessado em 03/07/2016

<https://www.ibm.com/support/knowledgecenter/pt->

[/SSFKSJ_8.0.0/com.ibm.mq.pro.doc/q003030_.htm](https://www.ibm.com/support/knowledgecenter/pt-SSFKSJ_8.0.0/com.ibm.mq.pro.doc/q003030_.htm) , acessado em 06/07/2016