Análise de Consumo Energético e Identificação de Ataques de Negação de Serviço em Computação em Névoa

Diogo V. M. Cruz[1], **Daniel F. Pigatto**[1], **Ana Cristina B. Kochem Vendramin**[1]

[1]Programa de Pós-Gaduação em Computação Aplicada (PPGCA)
Universidade Tecnológica Federal do Paraná (UTFPR)
Av. Sete de Setembro 3165, Rebouças 80.230-901 Curitiba - PR

diogocruz@alunos.utfpr.edu.br, {pigatto, criskochem}@utfpr.edu.br

Resumo. *O constante aumento na utilização de dispositivos de Internet das Coisas (IoT), fez crescer também a preocupação com segurança. Profissionais do ecossistema de IoT devem atentar-se à possibilidade de ataques cibernéticos, dentre eles o Ataque de Negação de Serviço Distribuído (DDoS). Esta modalidade de ataque pode representar não só um pequeno atraso nas comunicações, como também a interrupção total do serviço, seja pela inundação de pacotes na rede ou pela drenagem de recursos dos equipamentos. Tratando-se de dispositivos restritos, como os de IoT, qualquer pequena elevação em seu padrão de consumo energético pode representar uma alteração significativa em seu funcionamento. Através do uso da lógica Fuzzy, este trabalho propõe inferir o grau de pertinência de um ataque DDoS em um dispositivo central da computação em névoa por meio da análise do seu padrão de consumo energético.*

Abstract. *The constant increase in the use of Internet of Things (IoT) devices, has also raised concerns regarding security. Professionals in IoT ecosystem should be aware of the possibility of cyber attacks, such as the Distributed Denial of Service (DDoS). This type of attack can represent not only a small delay in communications, but also a total interruption of service, whether due to the flooding of packets in the network, or even the draining of equipment resources. In case of restricted devices, such as IoT devices, any small increase in their energy consumption pattern can represent a significant change in their functioning. By using Fuzzy logic, this work proposes to infer the pertinence of a DDoS attack in a central device of a fog computing through the analysis of its energy consumption pattern.*

1. Introdução

A computação pervasiva, caracterizada pela permanente interação entre o homem e os equipamentos de tecnologia da informação [Singh et al. 2014], abriu espaço para o surgimento da Internet das Coisas (IoT - *Internet of Things*), a qual tem assumido um papel importante nas atividades humanas.

Termo cunhado por Kevin Ashton no final dos anos 1990 [Kodali and Soratkal 2017], IoT refere-se a sensores ("coisas") conectados à rede que interagem com o mundo real. Englobando nichos específicos de aplicação, como a Indústria 4.0 [Peralta et al. 2017], o crescimento do ecossistema de IoT conta com estimativas de cerca de 75 bilhões de dispositivos conectados até 2050

[Shapsough et al. 2018]. Esse aumento no número de dispositivos conectados gera um gargalo nos meios de comunicações, devido à quantidade de troca de mensagens entre os nós [Peralta et al. 2017].

Uma medida introduzida pela Cisco com o objetivo de minimizar o problema de latência na rede e atraso no processamento foi a criação da computação em névoa, uma camada para análise de dados situada entre a nuvem e a borda da rede (dispositivos IoT) [Veeramanikanda.M 2017]. Um dos principais protocolos de comunicação utilizados na computação em névoa é o MQTT (*Message Queue Telemetry Transport*) [Shapsough et al. 2018]. Sua ampla utilização se deve ao fato de ser um protocolo leve, de fácil implementação e consumir poucos recursos computacionais dos dispositivos de IoT [Pavelic et al. 2018]. A arquitetura do protocolo MQTT requer, dentre outros componentes, um dispositivo central, chamado *broker*, responsável por gerenciar e rotear as comunicações entre processos de origem e destino [Pavelic et al. 2018]. Considerando um cenário de completa dependência do *broker* por parte do protocolo MQTT e a grande incidência de ataques cibernéticos a redes IoT [Harsha et al. 2018], destaca-se a relevância de se abordar segurança em redes IoT e, consequentemente, na computação em névoa.

Na computação em névoa observam-se diversas vulnerabilidades, tais como: a) Ataque de Negação de Serviço Distribuído (DDoS - *Distributed Denial of Service*) [Roohi et al. 2019, Haripriya and Kulothungan 2019, Shapsough et al. 2018]; b) Ataque de Inundação de mensagens CONNECT, o qual consiste em inundar o *broker* com diversas solicitações de conexão [Potrino et al. 2019]; c) Ataques de Privação de Sono (*Sleep Deprivation Attack*), através de execução de sub-rotinas maliciosas, forçando o nó a tratá-las mantendo-se sempre ativo [Shapsough et al. 2018].

O DDoS é o tipo de ataque mais recorrente e nocivo a dispositivos e redes IoT [Vignau et al. 2019]. Esse ataque visa exaurir recursos e causar indisponibilidade permanente ou temporária de um determinado serviço, ou somente gerar um atraso nas respostas a alguns clientes [Chen et al. 2018, Potrino et al. 2019]. Em um cenário de ataque DDoS, o *broker* é o principal alvo, uma vez que sua indisponibilidade afeta todo o sistema [Dinculeană and Cheng 2019]. Diante de um DDoS, os recursos energéticos de um *broker* podem ser drenados, contribuindo com sua inoperância de maneira prematura. Reforça-se, então, a necessidade de um método eficiente de detecção de ataques em *brokers* [Haripriya and Kulothungan 2019].

Alguns trabalhos na literatura propõem soluções que visam melhorar a segurança em ambientes de IoT.

Em [Chen et al. 2018] é conduzida uma análise de consumo de recursos de hardware em sistema de IoT quando sob ataque de Negação de Serviço. No artigo foram utilizadas ferramentas nativas do sistema operacional Kali Linux, como o *hping3* e o *nping*. Os testes foram realizados com diferentes tamanhos de pacotes com o objetivo de analisar o resultado em cada um deles. A eficiência do ataque foi medida pela latência do comando *ping* entre duas máquinas da rede, sendo uma delas o alvo.

Em [Bao et al. 2016] é apresentada uma ferramenta para testar a resiliência de dispositivos de IoT em ataques de negação de serviço. São medidos o consumo dos recursos de hardware (RAM, CPU e disco) e o tempo de atraso no tráfego de mensagens entre

clientes. Três tipos de ataques são realizados: o Ataque de Privação de Sono (*Sleep Deprivation Attack*) que drena o tempo de vida útil de um sensor alimentado por bateria; o ataque de Inferência de Radiofrequência (*Radio Frequency Inference*) que inunda o espectro com diversas frequências, causando interrupção no serviço; e o Ataque de Inundação MQTT (*MQTT Flooding Attack*) que bombardeia o *broker* com diversas mensagens de publicação e assinatura.

A abordagem encontrada em [Haripriya and Kulothungan 2019] identifica possíveis ataques DDoS em um *broker* por meio da análise do comportamento das trocas de mensagens durante o estabelecimento de uma conexão MQTT (CONNECT e CONNACK). O modelo de ameaça abordado visa identificar os ataques contra o *broker* analisando possíveis alterações no seu padrão de comportamento. A abordagem utiliza a lógica Fuzzy para identificar anomalias no comportamento dessas mensagens e inferir o grau de pertinência de um ataque DDoS.

Diferentes ataques de negação de serviço em dispositivos de IoT são apresentados em [Liang et al. 2017]. O experimento é realizado em uma rede local e conta com quatro componentes essenciais: um servidor, uma máquina virtual Kali Linux no computador atacante, um roteador Cisco e Arduino (Vítima). O consumo de recursos de hardware da vítima e a latência na rede são avaliados por meio de três tipos de ataques: o ataque de negação de serviço usando o hping3 com IP de origem aleatório, a inundação de pacotes SYN simples com IP (*Internet Protocol*) falsificado e a inundação de conexões TCP (*Transmission Control Protocol*). Ao final, as modalidades de ataque são analisadas e comparadas, sendo o ataque de negação de serviço usando o hping3 com IP de origem aleatório o mais nocivo dentre todos, gerando um elevado consumo de recursos e uma alta taxa de perda de pacotes.

O objetivo do presente trabalho é coletar e analisar informações sobre o padrão de consumo energético de um dispositivo central (*broker*) em computação em névoa, tanto durante seu funcionamento normal, bem como sob um ataque cibernético. Através dessa análise será possível identificar a ocorrência de um ataque DDoS e apresentar a intensidade com que este ataque ocorre, uma vez que quanto mais forte forem as investidas do atacante sobre o *broker* maior será o valor energético consumido pelo alvo. As informações coletadas servirão como entrada para um sistema de inferência Fuzzy que será capaz de inferir o grau de pertinência de um ataque de DDoS nesse dispositivo. A lógica Fuzzy foi escolhida com o objetivo de discretizar e apresentar com maior simplicidade e clareza o grau de pertinência de um ataque DDoS, se comparada à lógica binária convencional, apresentando com maior precisão os resultados obtidos e diminuindo a dependência de modelos matemáticos complexos [Shah 2018, El-Semary et al. 2005, Berouine et al. 2019].

Este trabalho está dividido em 5 seções. A seção 2 descreve a computação em névoa. A seção 3 apresenta a lógica Fuzzy. A seção 4 compreende a dinâmica do experimento e apresenta os resultados obtidos. Finalmente, a conclusão e trabalhos futuros são apresentados na seção 5.

2. Computação em Névoa

A Computação em névoa (*Fog computing*) é um conceito de tecnologia em camadas que une a borda da rede (dispositivos IoT) ao seu núcleo, a computação na nuvem, sendo a

névoa considerada uma extensão à nuvem [Peralta et al. 2017, Osanaiye et al. 2017].

O surgimento da computação em névoa se deve, em parte, justamente à falta de serviços de qualidade nos limites da rede para lidar com uma grande quantidade de dados de fontes distribuídas geograficamente [Osanaiye et al. 2017]. Logo, a computação em névoa aproxima a computação em nuvem dos dispositivos finais de IoT, realizando o processamento e análise dos dados mais próximo da origem e permitindo que ações sejam executadas mais rapidamente [Xu et al. 2016].

A arquitetura da computação em névoa consiste de três camadas. A camada mais extrema é composta por uma rede de "coisas" inteligentes, tais como sistemas e sensores embarcados. A camada intermediária é composta pelos nós de névoa, dispositivos com poder de análise e processamento. A camada mais centralizada é composta por centros de dados [Osanaiye et al. 2017].

Existem vários protocolos de camada de aplicação utilizados em comunicações IoT, os quais necessitam ser leves tendo em vista as restrições de recursos dos dispositivos [Kodali and Soratkal 2017]. Os protocolos mais usuais são: o MQTT, o *Hypertext Transfer Protocol* (HTTP), o *Constrained Application Protocol* (CoAP), o *Extensible Messaging and Presence Protocol* (XMPP) e o *Advanced Message Queuing Protocol* (AMQP) [Niruntasukrat et al. 2016].

Neste trabalho optou-se pelo uso do protocolo MQTT por ser um protocolo amplamente utilizado em comunicações de IoT, principalmente quando na utilização de dispositivos que possuem poucos recursos computacionais (com atuadores e sensores) [Pavelic et al. 2018, Vrettos et al. 2018].

2.1. Protocolo MQTT

O MQTT (*Message Queue Telemetry Transport*) é um protocolo leve da camada de aplicação que possui sobrecarga mínima de pacotes, é tolerante a redes não confiáveis, e é mais eficiente nas transmissões de dados em redes IoT, se comparado ao HTTP (*Hypertext Transfer Protocol*) [Naik 2017, Dinculeană and Cheng 2019].

O MQTT foi projetado para atuar em redes que dispõem de pouca largura de banda, garantindo algum grau de confiabilidade na entrega de mensagens pois funciona com base na pilha de protocolos TCP/IP (*Transmission Control Protocol/Internet Protocol*) [Naik 2017, Lekić et al. 2019].

O protocolo MQTT possui três níveis de Qualidade de Serviço (QoS - *Quality of Service*) [Toldinas et al. 2019]:

- ***At most once:*** "No máximo uma vez", conhecido como *"fire and forget"*. É o nível de QoS padrão do MQTT e é representado pelo número "0";
- ***At least once:*** "Ao menos uma vez". O nível de QoS 1 garante que uma mensagem seja entregue pelo menos uma vez ao destinatário. É possível que uma mensagem seja enviada ou entregue várias vezes;
- ***Exactly once:*** "Exatamente uma vez". O nível de QoS 2 garante que a mensagem seja entregue exatamente uma vez, com envio de confirmações de recebimento.

2.2. Sistema de Publicação e Assinatura

A comunicação entre os processos na computação em névoa é regida por um sistema chamado publicação e assinatura (*Publish and Subscribe*), o qual exige a presença de três

atores essenciais [Xu et al. 2016]:

- *Broker*: ponto central em uma computação em névoa, sendo o responsável por conectar os nós da IoT à plataforma em nuvem e por receber todas as mensagens, filtrá-las e roteá-las aos dispositivos finais [Dinculeană and Cheng 2019, Venanzi et al. 2018]. O *broker* serve como um serviço de mensagens, o qual recebe e repassa as mensagens por meio de tópicos que são estruturas (*strings*) hierárquicas que alocam e filtram a distribuição de mensagens [Pavelic et al. 2018]. Os clientes necessitam referenciar algum tópico durante o processo de publicação ou assinatura [Shinde et al. 2016]. Sem o *broker*, a troca de mensagens na computação em névoa fica completamente impossibilitada, sendo um dispositivo indispensável [Dinculeană and Cheng 2019, Shinde et al. 2016];
- Publicador: processo responsável por transmitir os dados coletados ao *broker*;
- Assinante: processo que tem interesse em tópicos específicos de dados. Ele recebe do *broker* os dados do seu interesse.

Como pode ser visto na Figura 1, ambos publicador e assinante são completamente desacoplados. A responsabilidade de manipular as comunicações e gerar o sincronismo entre esses dois processos distintos e desconhecidos entre si é do *broker*.

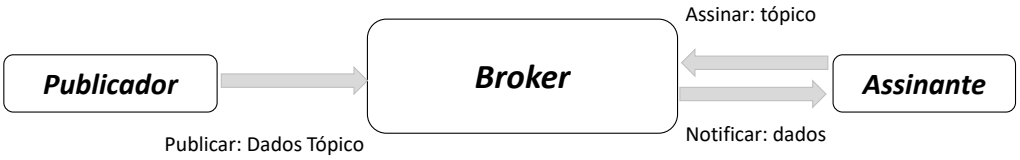


Figura 1. Sistema de Publicação e Assinatura. Adaptado de [Lekić et al. 2019]

3. Lógica Fuzzy

A estrutura básica de um sistema Fuzzy, conforme pode ser visto na Figura 2, consiste em [Rahimi and Chrysostomou 2019]:

- **Unidade de fuzzificação**: esta unidade consiste nos valores de entrada do sistema. O valor de cada entrada é então avaliado, obtendo-se como resultado o grau de pertinência de cada uma das entradas por meio de um conjunto de variáveis linguísticas, as quais estão compreendidas em um universo de discurso. O universo de discurso representa o intervalo de valores mínimo e máximo que uma variável de entrada pode assumir [Shah 2018];
- **Base de regras**: é composta por regras definidas de maneira empírica por um especialista e são combinadas por instruções do tipo *SE – ENTÃO* [Rahimi and Chrysostomou 2019, Shah 2018];
- **Mecanismo de Inferência ou Unidade de raciocínio**: o raciocínio da lógica Fuzzy é muito próximo do pensamento humano, se comparada a técnicas tradicionais. Na lógica clássica, um cenário só pode ser entendido como falso ou verdadeiro (0 ou 1). Através de Fuzzy, uma determinada grandeza pode ser representada por valores discretos dependendo de um ponto de vista. Assim como no mundo

real, em Fuzzy a representação poder ser feita de maneira escalonada, inferindo o quanto de verdadeiro ou de falso possui um resultado [Berouine et al. 2019]. No mecanismos de inferência fuzzy os seguintes passos devem ser seguidos [Pappis and Siettos 2005]: *matching* e agregação dos antecedentes, aplicação da semântica da regra e defuzzificação. No *matching* dos antecedentes, o sistema Fuzzy insere o valor de cada entrada no universo de discurso de suas respectivas variáveis linguísticas, o qual avalia o grau de pertinência de cada entrada neste universo. Cada grau de pertinência gerado por esses antecedentes são combinados entre si por meio de um operador de composição, o qual determinará o valor de saída desta agregação. Um operador de composição comum é o *max-min*, o qual obtém o valor máximo dos mínimos das combinações de grau de pertinência dos antecedentes. A semântica da regra é atribuída pela conjunção de Mamdani, a qual é dada pelo operador *min()* [Mamdani and Assilian 1975]. A escolha por Mamdani deve-se ao fato desta ser a semântica da regra mais presente na literatura [Haripriya and Kulothungan 2019, Rahimi and Chrysostomou 2019], bem como promove a ativação apenas das regras com grau de pertinência maiores que zero. Para cada regra é computado o valor inferido correspondente, baseado na agregação dos antecedentes e na semântica da regra escolhida. Na etapa inicial de defuzzificação, a cargo do mecanismo de inferência, o resultado obtido na saída é um conjunto Fuzzy. Esse conjunto não é ainda um resultado de fácil interpretação. Este conjunto de saída alimentará o estágio final de defuzzificação, descrito a seguir, no qual obtém-se uma saída única [Santiago and Arockiam 2017].
- **Unidade de defuzzificação**: nesta fase é ponderado e combinado vários conjuntos Fuzzy resultantes do processo de inferência [Shah 2018]. Esses conjuntos Fuzzy servem de entrada a um segundo estágio de defuzzificação, pois, diferentemente do observado no mecanismo de inferência, esta defuzzificação não retorna um conjunto Fuzzy e sim um valor único, denominado valor crisp [Santiago and Arockiam 2017]. Os métodos de defuzzificação mais comumente usados são os da média dos máximos, o centro da soma das áreas e centróide [Lee 1990], sendo este último utilizado no presente trabalho. O método centróide converte o centro da gravidade de um conjunto Fuzzy de saída em um valor único (crisp), facilitando a leitura do resultado.

4. Resultados e discussões

Esta seção apresenta os experimentos realizados para inferir o grau de pertinência de um ataque de negação de serviço distribuído, baseado no comportamento do consumo energético em um *broker* MQTT, utilizando regras de um sistema de inferência Fuzzy.

4.1. Experimento

O experimento consiste no envio de dados colhidos por um sensor de temperatura DHT11 (publicador) para o *broker*, que os repassa a um *smartphone* (assinante). O broker MQTT utilizado neste trabalho é o Mosquitto, uma implementação leve e frequentemente usada em associação ao microprocessador Raspberry Pi [Lekić et al. 2019, Pavelic et al. 2018]. Os componentes do experimentos estão apresentados na Tabela 1.

Durante o sistema de publicação e assinatura, a corrente elétrica é lida pelo sensor de corrente ACS712. Este sensor foi escolhido por sua alta precisão e por possuir o

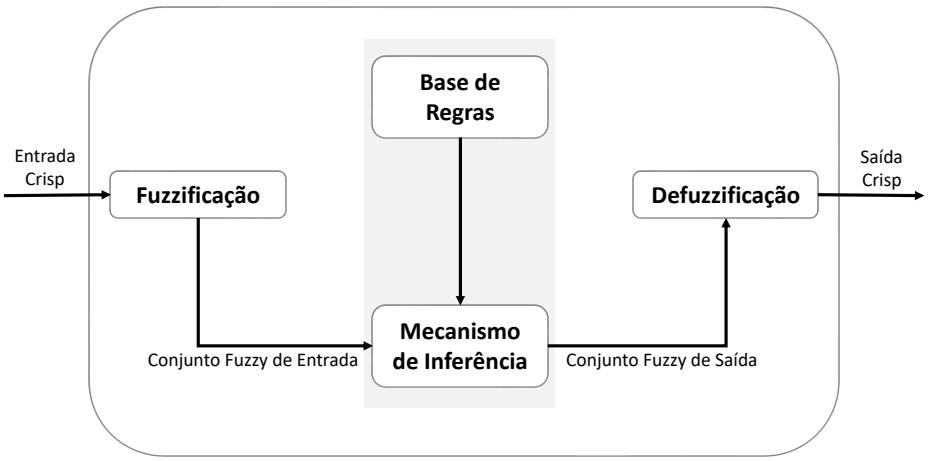


Figura 2. Sistema Fuzzy. Adaptado de [Santiago and Arockiam 2017]

Tabela 1. Componentes do Experimento

Dispositivo	**Função**	**Software**
Raspberry Pi 3	*Broker*	Raspbian Stretch 4.14
DHT 11 (sensor temperatura)	Publicador	-
Smartphone (Android 9.0)	Assinante	MQTT Dashboard v1.9.3
Notebook Dell Inspirion 15	Atacante	(VM) Kali Linux 64 Bits
Sensor de Corrente ACS712	Medidor de corrente	-
Arduino Uno	Microcontrolador	-

melhor custo benefício entre os sensores disponíveis no mercado [Khwanrit et al. 2018]. A corrente lida é manipulada por um código embarcado no Arduino. A média do consumo de corrente elétrica durante um determinado período de tempo é então considerada como entrada do Sistema de Inferência Fuzzy. O fluxo de mensagens segue o nível de QoS 0 do MQTT, pois o objetivo é medir o consumo energético em condições mínimas de utilização [Peralta et al. 2017], uma vez que níveis mais elevados de QoS consomem mais energia [Shapsough et al. 2018].

Durante o funcionamento normal do *broker*, a média do consumo de corrente elétrica e lidas no Raspberry foram de 375 mAh e 1,98 W, respectivamente. A Figura 3 apresenta um comparativo do consumo energético de um *broker* durante seu funcionamento normal e quando sob ataque DDoS. Este experimento teve seu início imediatamente antes do *boot* do sistema. A leitura foi realizada durante 120 segundos e o ataque é iniciado no segundo 60. O processo de inicialização do Raspberry consome um alto valor energético absoluto [Bekaroo and Santokhee 2016], elevando os dados de leitura a patamares semelhantes aos observados durante um ataque DDoS. Por esse motivo, deve-se levar em consideração o intervalo de tempo em que esse consumo ocorre.

O ataque ao *broker* é caracterizado por explorar uma técnica conhecida como inundação de pacotes SYN [Roohi et al. 2019], com pacotes de 100 bytes de *payload* enviados ao *broker*, na porta padrão do MQTT (1883), utilizando endereços aleatórios de IP de origem. Esta inundação eleva o patamar de consumo energético do nível médio

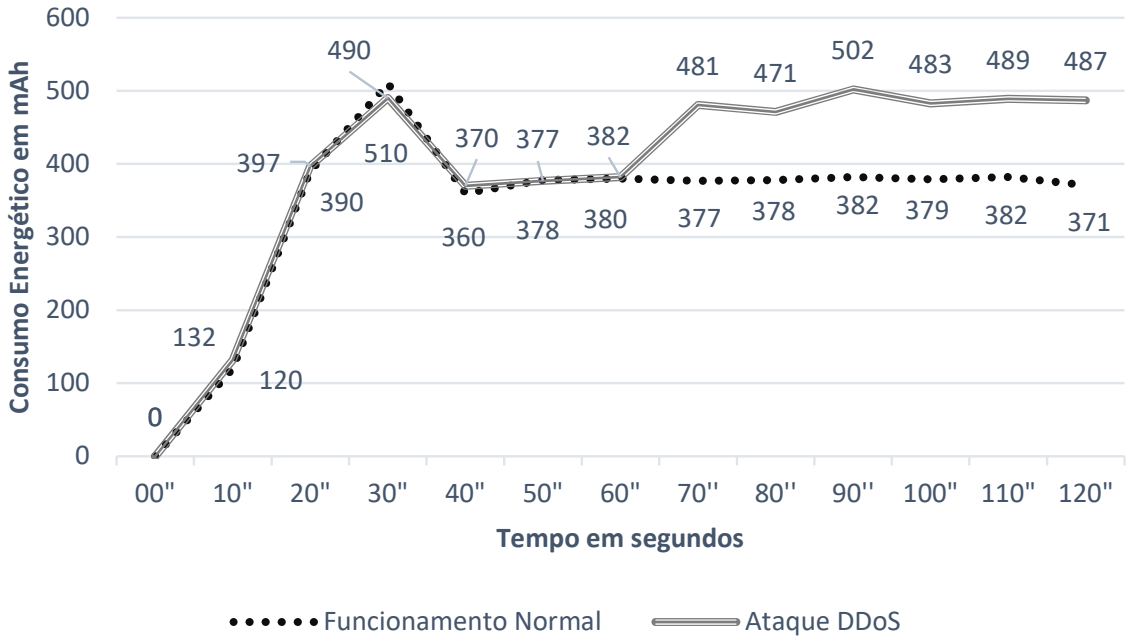


Figura 3. Consumo Energético do *Broker*

(CM - Consumo Médio) para o nível alto (CA - Consumo Alto), chegando a correntes que ultrapassam o valor de 510 mAh com potência de 2,7 W, o que eleva em 36% o consumo energético. Mantendo-se neste patamar a maior parte do tempo, caracteriza-se um ataque.

A ferramenta de ataque utilizada foi o *hping3*, um utilitário nativo do Kali Linux [Chen et al. 2018]. O ataque foi gerado a partir da seguinte instrução:

"hping3 -c 100 -d 100 -S -p 1883 –flood –rand-source 10.41.52.100",

onde "-c 100" representa a quantidade de pacotes desferidos à máquina alvo, "-d 100" indica o tamanho dos pacotes em bytes. Com o parâmetro "-S" o alvo recebe somente a flag SYN na porta padrão do MQTT informada em "-p 1883". O item *"- -flood"* projeta o ataque a uma velocidade mais rápida possível, sem mostrar qualquer resposta. O endereço de IP de origem é aleatório por meio do parâmetro *"- -rand-source"*.

4.2. Sistema de Inferência Fuzzy

No presente trabalho, a implementação do sistema de inferência Fuzzy é realizada em Python com a biblioteca *fuzzy logic toolbox* e são utilizadas as seguintes variáveis linguísticas como base de dados:

- **Consumo Energético**: T(X1) = CB (Consumo Baixo), CM (Consumo Médio), CA (Consumo Alto).
- **Tempo**: T(X2) = TMC (Tempo Muito Curto), TC (Tempo Curto), TM (Tempo Médio), TL (Tempo Longo), TML (Tempo Muito Longo).
- **Grau de Pertinência de Ataque**: T(Y) = AB (nível de Ataque Baixo), AM (nível de Ataque Médio), AA (nível de Ataque Alto).

A Tabela 2 apresenta a base de regras envolvendo as variáveis linguísticas supracitadas, onde a construção do conjunto de regras Fuzzy deve ser extraído mediante o

cruzamento dos índices.

Tabela 2. Base de Regras proposta para o Sistema de Inferência Fuzzy

ÍNDICES	CB	CM	CA
TMC	AB	AB	AM
TC	AB	AB	AM
TM	AB	AM	AA
TL	AB	AM	AA
TML	AB	AM	AA

Neste trabalho a semântica da regra utilizada é a conjunção de Mamdani. O processo de defuzzificação adota a abordagem centróide, na qual a área preenchida pelo gráfico é convertida em um único valor de saída.

O conjunto de regras é definido por instruções *"Se antecedente - Então consequente")*. Como exemplo da primeira regra (R1), tem-se:

- R1: Se X1 é CB E X2 é TMC Então Y é AB

Os valores da média do consumo energético e tempo formam as entradas do sistema Fuzzy. Os valores do universo de discurso "Consumo de Energia" estão entre 0 e 600 mAh, representados com os seguintes intervalos (variáveis linguísticas): CB [0, 0, 300], CM [250, 375, 410] e CA [340, 600, 600]. O universo de discurso "Tempo"˜ é representado com os valores de 0 a 60 segundos, sendo definidos com os seguintes intervalos: TMC [0, 0, 15], TC [0, 15, 30], TM [15, 30, 45] TL [30, 45, 60] e TML [45, 60, 60]. Em ambos os universos de discurso é adotada a função Triangular como função de pertinência, a qual é computacionalmente mais simples, possui intervalos com limites estabelecidos e valores que variam de maneira linear [Matam et al. 2017].

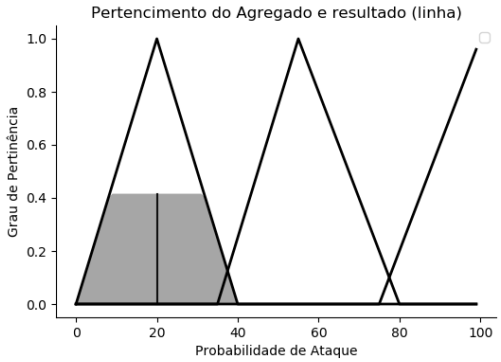
A Tabela 3 apresenta três cenários onde a média do consumo energético, a potência e o tempo decorrido são coletados, servindo de dados de entrada para o sistema de inferência Fuzzy.

Tabela 3. Cenários de Ataque a um *Broker* MQTT

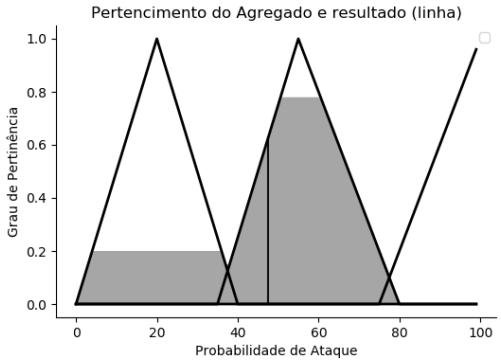
Cenário de Ataque	Média de Consumo (mAh)	Potência (W)	Tempo (s)	Nível de Ataque
(1)	489	2,5	7	Ataque Baixo
(2)	383	2,02	57	Ataque Médio
(3)	490	2,6	53	Ataque Alto

A Figura 4 mostra o gráfico resultante dos cenários supracitados, os quais exemplificam três graus de pertinência de ataque DDoS. Para análise desses graus de pertinência, considera-se um consumo normal de operação de 375 mAh e 1,98W, conforme apresentado na Figura 3, e um tempo de 30 segundos representando o tempo necessário para o *boot* do sistema. No cenário (1), durante 7 segundos a média de consumo foi de 489 mAh e a potência de 2,5 W, o que é considerado uma média de consumo alta em um *broker*, pois está bem acima do consumo normal de operação. Entretanto, levando-se em conta o tempo decorrido de apenas 7 segundos, o qual é muito curto, é determinado um grau

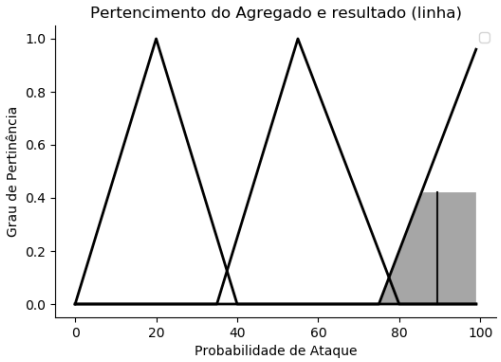
de pertinência baixo de ataque (AB). No cenário (2), a média de consumo aferida está um pouco acima do consumo normal de operação, tendo como resultado um grau de pertinência médio de ataque (AM). No cenário (3), a média de consumo é considerada alta, muito acima dos padrões normais de operação do *broker*, e se mantém por um período de tempo maior que 30 segundos, o que caracteriza um grau de pertinência alto de ataque (AA).



(a) Nível de Ataque Baixo



(b) Nível de Ataque Médio



(c) Nível de Ataque Alto

Figura 4. Grau de Pertinência de um Ataque de Negação de Serviço Distribuído

5. Conclusões e Trabalhos Futuros

O ataque de negação de serviço distribuído é o tipo de ataque mais recorrente e nocivo a dispositivos na computação em névoa. Este tipo de ataque pode representar não só

um pequeno atraso nas comunicações, como também a interrupção permanente ou temporária de um determinado serviço, como o serviço de mensagens providos por um *broker*. Tratando-se de um dispositivo com recursos escassos, qualquer pequena elevação em seu padrão de consumo energético pode representar uma alteração significativa em seu funcionamento.

Este trabalho objetivou medir o consumo energético de um *broker* em computação em névoa durante seu funcionamento normal e sob um ataque de negação de serviço. O objetivo foi mapear seu padrão de consumo energético e inferir, baseado nos dados coletados, o grau de pertinência de um ataque de DDoS. A lógica Fuzzy foi utilizada neste processo, pois apresenta com maior precisão o grau de pertinência de cada valor coletado, se comparada à lógica tradicional.

Nos testes realizados foi observado que o *boot* do sistema e um ataque DDoS possuem picos energéticos semelhantes. Portanto, foi detectada a necessidade de levar em consideração o tempo decorrido em um determinado consumo de energia, pois o processo completo de inicialização do Raspberry (*broker*) não dura mais que 30 segundos, já um ataque de DDoS pode durar horas. O relacionamento das variáveis antecedentes "tempo" e "consumo" permitiu inferir o grau de pertinência de um ataque de negação de serviço distribuído, analisando apenas o comportamento energético de um *broker*. Outro ponto a ser destacado é que mesmo o serviço de troca de mensagens podendo não parar completamente durante o ataque, os recursos dele serão esgotados prematuramente, uma vez que o ataque foi capaz de elevar o consumo energético do *broker* em cerca de 36%.

Como trabalhos futuros, pretende-se identificar outros tipos de ataques em um dispositivo de IoT avaliando seu consumo energético.

Referências

Bao, C., Guan, X., Sheng, Q., Zheng, K., and Huang, X. (2016). A Tool for Denial of Service Attack Testing in IoT. page 5.

Bekaroo, G. and Santokhee, A. (2016). Power consumption of the Raspberry Pi: A comparative analysis. *2016 IEEE International Conference on Emerging Technologies and Innovative Business Practices for the Transformation of Societies, EmergiTech 2016*, pages 361–366.

Berouine, A., Akssas, E., Naitmalek, Y., Lachhab, F., Bakhouya, M., Ouladsine, R., and Essaaidi, M. (2019). A Fuzzy Logic-Based Approach for HVAC Systems Control. pages 1510–1515.

Chen, Q., Chen, H., Cai, Y., Zhang, Y., and Huang, X. (2018). Denial of Service Attack on IoT System. *Proceedings - 9th International Conference on Information Technology in Medicine and Education, ITME 2018*, pages 755–758.

Dinculeană, D. and Cheng, X. (2019). Vulnerabilities and Limitations of MQTT Protocol Used between IoT Devices. *Applied Sciences*, 9(5):848.

El-Semary, A., Edmonds, J., Gonzalez, J., and Papa, M. (2005). A framework for hybrid fuzzy logic intrusion detection systems. *IEEE International Conference on Fuzzy Systems*, pages 325–330.

Haripriya, A. P. and Kulothungan, K. (2019). Secure-MQTT: an efficient fuzzy logic-based approach to detect DoS attack in MQTT protocol for internet of things. *EURASIP Journal on Wireless Communications and Networking*, 2019(1):90.

Harsha, M. S., Bhavani, B. M., and Kundhavai, K. R. (2018). Analysis of vulnerabilities in MQTT security using Shodan API and implementation of its countermeasures via authentication and ACLs. *2018 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2018*, pages 2244–2250.

Khwanrit, R., Kittipiyakul, S., Kudtonagngam, J., and Fujita, H. (2018). Accuracy Comparison of Present Low-cost Current Sensors for Building Energy Monitoring. *2018 International Conference on Embedded Systems and Intelligent Technology and International Conference on Information and Communication Technology for Embedded Systems, ICESIT-ICICTES 2018*, pages 3–8.

Kodali, R. K. and Soratkal, S. R. (2017). MQTT based home automation system using ESP8266. *IEEE Region 10 Humanitarian Technology Conference 2016, R10-HTC 2016 - Proceedings*, pages 1–5.

Lee, C. (1990). Fuzzy logic in control systems: fuzzy logic controller. II. *IEEE Transactions on Systems, Man, and Cybernetics*, 20(2):419–435.

Lekić, M., Galić, J., and Matić, S. (2019). An IoT Solution for Secured and Remote Sound Level Monitoring. *2019 18th International Symposium INFOTEH-JAHORINA, INFOTEH 2019 - Proceedings*, (March):20–22.

Liang, L., Zheng, K., Sheng, Q., and Huang, X. (2017). A denial of service attack method for an IoT system. *Proceedings - 2016 8th International Conference on Information Technology in Medicine and Education, ITME 2016*, pages 360–364.

Mamdani, E. H. and Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, 7(1):1–13.

Matam, R., Shukla, S., and Tyagi, G. (2017). Local Connectivity Based Boundary Detection in Wireless Sensor Networks. *Proceedings - 2016 IEEE International Conference on Internet of Things; IEEE Green Computing and Communications; IEEE Cyber, Physical, and Social Computing; IEEE Smart Data, iThings-GreenCom-CPSCom-Smart Data 2016*, pages 420–423.

Naik, N. (2017). Choice of Effective Messaging Protocols for IoT Systems : MQTT , CoAP , AMQP and HTTP.

Niruntasukrat, A., Issariyapat, C., Pongpaibool, P., Meesublak, K., Aiumsupucgul, P., and Panya, A. (2016). Authorization mechanism for MQTT-based Internet of Things. *2016 IEEE International Conference on Communications Workshops, ICC 2016*, pages 290–295.

Osanaiye, O., Chen, S., Yan, Z., Lu, R., Choo, K. K. R., and Dlodlo, M. (2017). From Cloud to Fog Computing: A Review and a Conceptual Live VM Migration Framework. *IEEE Access*, 5:8284–8300.

Pappis, C. P. and Siettos, C. I. (2005). *Fuzzy Reasoning*, pages 437–474. Springer US, Boston, MA.

Pavelic, M., Bajt, V., and Kusek, M. (2018). Energy efficiency of machine-to-machine protocols. *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2018 - Proceedings*, pages 361–366.

Peralta, G., Iglesias-Urkia, M., Barcelo, M., Gomez, R., Moran, A., and Bilbao, J. (2017). Fog computing based efficient IoT scheme for the Industry 4.0. *Proceedings of the 2017 IEEE International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics, ECMSM 2017*, pages 1–6.

Potrino, G., De Rango, F., and Santamaria, A. F. (2019). Modeling and evaluation of a new IoT security system for mitigating DoS attacks to the MQTT broker. *IEEE Wireless Communications and Networking Conference, WCNC*, 2019-April:1–6.

Rahimi, P. and Chrysostomou, C. (2019). Improving the Network Lifetime and Performance of Wireless Sensor Networks for IoT Applications Based on Fuzzy Logic. In *Proceedings - 15th Annual International Conference on Distributed Computing in Sensor Systems, DCOSS 2019*, pages 667–674. IEEE.

Roohi, A., Adeel, M., and Shah, M. A. (2019). DDoS in IoT: A roadmap towards security countermeasures. *ICAC 2019 - 2019 25th IEEE International Conference on Automation and Computing*, (September):1–6.

Santiago, S. and Arockiam, L. (2017). A novel fuzzy based energy efficient routing for Internet of Things. *2017 International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies, ICAMMAET 2017*, 2017-Janua:1–4.

Shah, B. (2018). Fuzzy Energy Efficient Routing for Internet of Things (IoT). *International Conference on Ubiquitous and Future Networks, ICUFN*, 2018-July:320–325.

Shapsough, S., Aloul, F., and Zualkernan, I. A. (2018). Securing Low-Resource Edge Devices for IoT Systems. *2018 International Symposium in Sensing and Instrumentation in IoT Era, ISSI 2018*.

Shinde, S. A., Nimkar, P. A., Singh, S. P., Salpe, V. D., and Jadhav, Y. R. (2016). MQTT - Message queuing telemetry transport. *International Journal of Research*, 3(3):240–244.

Singh, D., Tripathi, G., and Jara, A. J. (2014). A survey of Internet-of-Things: Future vision, architecture, challenges and services. *2014 IEEE World Forum on Internet of Things, WF-IoT 2014*, pages 287–292.

Toldinas, J., Lozinskis, B., Baranauskas, E., and Dobrovolskis, A. (2019). MQTT Quality of Service versus Energy Consumption. *2019 23rd International Conference Electronics*, pages 1–4.

Veeramanikanda.M, S. S. (2017). Publish/Subscribe Broker based Architecture for fog computing. International Conference on Energy, Communication, Data Analytics and Soft Computing.

Venanzi, R., Kantarci, B., Foschini, L., and Bellavista, P. (2018). MQTT-Driven Node Discovery for Integrated IoT-Fog Settings Revisited: The Impact of Advertiser Dynamicity. *Proceedings - 12th IEEE International Symposium on Service-Oriented System*

Engineering, SOSE 2018 and 9th International Workshop on Joint Cloud Computing, JCC 2018, pages 31–39.

Vignau, B., Khoury, R., and Halle, S. (2019). 10 Years of IoT Malware: A Feature-Based Taxonomy. *Proceedings - Companion of the 19th IEEE International Conference on Software Quality, Reliability and Security, QRS-C 2019*, pages 458–465.

Vrettos, G., Logaras, E., and Kalligeros, E. (2018). Towards Standardization of MQTT-Alert-based Sensor Networks: Protocol Structures Formalization and Low-End Node Security. *2018 IEEE 13th International Symposium on Industrial Embedded Systems, SIES 2018 - Proceedings*, pages 1–4.

Xu, Y., Mahendran, V., and Radhakrishnan, S. (2016). Towards SDN-based fog computing: MQTT broker virtualization for effective and reliable delivery. *2016 8th International Conference on Communication Systems and Networks, COMSNETS 2016*, pages 1–6.