IoTFogSim: Um Simulador Orientado a Eventos para Avaliação de Aplicações baseadas em IoT-Fog-Cloud

Rafael Sampaio Pereira[1], **Cássio Vinicius Serafim Prazeres**[1], **Matheus Thiago Marques Barbosa**[1], **Eric Bernardes Chagas Barros**[1], **Maycon Leone Maciel Peixoto**[1]

[1]Programa de Pós-Graduação em Ciência da Computação(PGCOMP)
Instituto de Matemática – Universidade Federal da Bahia (UFBA)
Salvador – BA – Brasil

{rafaspereira, matheus.thiago, eric.bernardes, prazeres}@ufba.br,

mayconleone@dcc.ufba.br

Abstract. *IoT technologies have provided new solutions through the use of a large amount of smart devices in multiple domains. Thus, researchers are proposing applications to handle the data generated in various areas such as healthcare, energy control, and home automation. On the other hand, due to cost and scalability constraints, testing and evaluating these applications in real-world scenarios is a complex and expensive task. Simulation is shown as an effective technique to validate computer networks. This paper presents IoTFogSim, an event-driven simulator that allows developers to simulate IoT-Fog-Cloud-based applications, protocols, and algorithms.*

Resumo. *As tecnologias IoT tem fornecido novas soluções através da utilização de uma grande quantidade de dispositivos inteligentes em múltiplos domínios. Dessa forma, os pesquisadores estão propondo aplicações para lidar com os dados gerados em diversas áreas como a saúde, controle de energia e a domótica. Por outro lado, devido a limitações de custos e escalabilidade, testar e avaliar estas aplicações em cenários do mundo real é uma tarefa complexa e dispendiosa. A simulação se mostra como técnica eficaz para validar redes de computadores. Este artigo apresenta o IoTFogSim, um simulador orientado a eventos que permite aos desenvolvedores simular aplicações, protocolos e algoritmos baseados em IoT-Fog-Cloud.*

1. Introdução

A *Internet* das Coisas (IoT) é um conceito associado a "*Internet* do Futuro", onde objetos do mundo real tornam-se parte da *internet* [Kraijak and Tuwanut 2015]. A IoT permite que objetos dispersos geograficamente enviem e recebam dados de áreas como saúde, controle residencial e redes elétricas. A quantidade de dispositivos IoT que produzem dados constantemente pode chegar a ser da ordem de biliões [Shah and Venkatesan 2018].

A computação em nuvem tem a capacidade de processamento ideal para lidar com grandes volumes de dados [Patel et al. 2019]. Além disso, o modelo baseado na nuvem tornou-se a abordagem padrão para a IoT [Coutinho et al. 2017], porém requer que todos os dados sejam enviados pela internet, o que leva a problemas como congestionamento da

rede de comunicação [Ravi Teja et al. 2015]. Dependendo da aplicação, não é possível utilizar a nuvem para tratar esses dados em tempo real [Katal et al. 2013]. A computação em névoa permite a execução de serviços da nuvem diretamente na borda da rede, proporcionando menor tráfego de rede, baixa latência e computação distribuída em tempo real [Peixoto et al. 2021]. Esse paradigma se mostra ideal para sistemas onde existe a coleta de dados de forma distribuída [Naranjo et al. 2016].

Validar aplicações IoT em um cenário do mundo real é uma tarefa custosa e de difícil execução, uma vez que requer um grande número de dispositivos [Zeng et al. 2017]. Em redes de computadores, a simulação é uma técnica comum para análise da redes, onde o comportamento dos nós é executado em um ambiente virtual.

As simulações para IoT precisam ser escaláveis, suportando muitos dispositivos, o que pode ser um problema se apenas um *host* (e.g. computador) executar todo o processo. As ferramentas de simulação baseadas em um único CPU não são capazes de escalar o grande número de nós requerido pela IoT [D'Angelo et al. 2016]. Faz-se necessária uma ferramenta de simulação que utilize múltiplos *hosts*.

Este artigo apresenta o IoTFogSim, um simulador orientado a eventos que utiliza o paradigma da Simulação Paralela e Distribuída (PADS) para permitir desenvolvedores de sistemas IoT, névoa e nuvem possam validar suas aplicações antes de construir a solução no mundo real. O restante deste artigo está estruturado da seguinte maneira. A Seção 2 apresenta trabalhos relacionados. O projeto e arquitetura da ferramenta de simulação proposta são descritos na Seção 3. A Seção 4 mostra os testes e a avaliação de desempenho da solução proposta. A Seção 5 apresenta os experimentos e a análise de resultados. Finalmente, a Seção 6 mostra considerações finais e trabalhos futuros.

2. Trabalhos Relacionados

A simulação é um método de alta relevância para a validação de redes de computadores. A fim de simular cenários de aplicações IoT, vários trabalhos apresentam diferentes soluções que implementam ferramentas de simulação.

[Han et al. 2014] apresentam um simulador multiplataforma chamado DPWSim, desenvolvido em Java. A ferramenta é baseada no Perfil de Dispositivos para Serviços Web (DPWS), um padrão que estabelece serviços Web para dispositivos computacionais limitados. Além disso, os usuários podem escolher entre utilizar dispositivos padrão ou criar novos dispositivos. Este simulador inclui interface gráfica de usuário (GUI)

Os resultados de um estudo proposto por [Gupta et al. 2016] mostram o iFogSim, uma ferramenta para gestão de recursos na nevoa, capaz de simular dispositivos finais, data centers na nuvem, e *links* de rede e que se mostra escalável para o contexto de IoT. Este simulador inclui uma GUI.

[Pflanzner et al. 2016] propõem o MobIoTSim, um simulador que foi desenvolvido no formato de aplicativo móvel para android onde múltiplos dispositivos e sensores podem ser simulados. A proposta possui um sistema na nuvem que é responsável por receber os dados vindos dos dispositivos simulados e que atua como um *gateway*, no qual o usuário pode visualizar os dados recebidos [Kertesz et al. 2018]. O simulador suporta até centenas de dispositivos. Este simulador inclui uma GUI.

O simulador IOTSim é apresentado em [Zeng et al. 2017] como uma extensão

para o simulador CloudSim. Essa ferramenta permite simular o processamento de *big data* e utiliza o algoritmo MapReduce para analisar os dados produzidos pelos dispositivos simulados. Este simulador não inclui uma GUI.

[Puliafito et al. 2020] apresentam o MobFogSim, uma extensão do iFogSim que permite a modelagem da mobilidade de dispositivos e a migração de serviços na computação em nevoa. Experimentos foram realizados levando em conta os padrões de mobilidade de um usuário, derivados do Tráfego de Luxemburgo no SUMO (LuST).

[Qayyum et al. 2018] apresentam o FogNetSim++, uma solução para simulação de sistemas computacionais que inclui uma GUI e suporta dispositivos IoT. Esse simulador é uma extensão para o OMNET ++. Os protocolos fornecidos para comunicação na nevoa são: MQTT, CoAP, e MQP. Essa ferramenta permite que o usuário incorpore seus próprios modelos de mobilidade e algoritmos de agendamento nos nós da nevoa.

3. Simulador Proposto

Simulação Paralelas e Distribuídas (PADS) é um paradigma onde as simulações de rede são realizadas por múltiplos processadores interconectados [Perumalla 2006]. Escalabilidade e interoperabilidade estão entre as razões para utilizar as simulações PADS [D'Angelo et al. 2016]. Este artigo descreve o projeto e implementação de um simulador distribuído orientado a eventos baseado no paradigma PADS chamado de IoTFogSim[1] [2]. O principal objetivo do simulador proposto é permitir que algoritmos e protocolos para IoT possam ser avaliados. A Figura 1 mostra a tela principal do simulador.

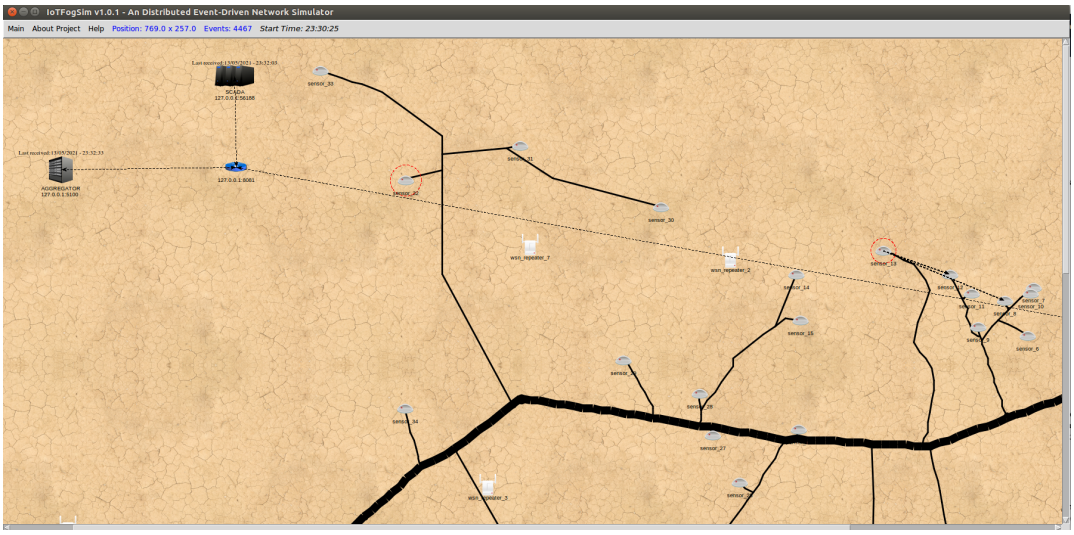

Figure 1. Tela principal do IoTFogSim

3.1. Arquitetura do Simulador

O simulador IoTFogSim foi desenvolvido utilizando a linguagem de programação Python na versão 3.5. A fim de construir uma ferramenta que reage a eventos, foram utilizados

[1]Video de apresentação disponível em `https://youtu.be/9sOloBC_PpI`

[2]Código fonte e manual do usuário disponíveis em `https://github.com/Rafael-Sampaio-Pereira/IoTFogSim`

o *framework* de rede Twisted e a biblioteca gráfica Tkinter. O Twisted permite que as aplicações possam reagir a eventos de rede executando ações em respostas a esses eventos [Labs and Twisted Matrix 2019]. A biblioteca Tkinter permite que as aplicações possam reagir a eventos da tela (e.g. colisão e sobreposição) [Shipman 2013].

3.2. Arquitetura dos Nós

Todos os nós do simulador, exceto os nós da rede de sensores, são representados no simulador IoTFogSim como uma instancia da classe *DeviceComponent*, que é composta por outras três classes: *VisualComponent*, *ApplicationComponent* e *NetworkComponent*.

- ***VisualComponent*** - Classe baseada na biblioteca Tkinter que executa as tarefas gráficas, onde o usuário pode customizar o ícone de um nó. Nessa classe são armazenadas as dimensões e coordenadas dos ícones dos nós em relação a tela.
- ***ApplicationComponent*** - Classe baseada no *framework* Twisted que contem a camada de aplicação do nó simulado. Nessa classe o usuário pode colocar seus algoritmos e protocolos personalizados.
- ***NetworkComponent*** - Classe baseada no *framework* Twisted que representa a camada de rede do nó simulado. Essa classe é utilizada para ativar *sockets* de rede reais e conectar um nó a outros através dos protocolos SSL, TLS, TCP/IP ou UDP.

Os nós da rede de sensores sem fio não utilizam *sockects* reais e suas características de camada física e rede podem ser configuradas pelo usuário sendo líder o único nó que possui um *socket* real. A Figura 2 mostra a estrutura de um nó no IoTFogSim.

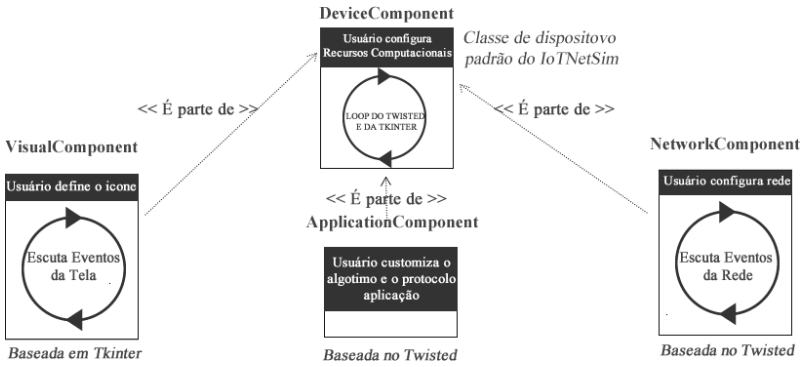

Figure 2. Arquitetura de um nó do simulador IoTFogSim.

3.3. Arquitetura das Simulações

Os usuários podem avaliar os algoritmos e protocolos das suas aplicações IoT simulando o seu cenário de maneira centralizada em único *host* ou distribuída. Simulações com poucos dispositivos e que não produzirão dados em grandes formatos, podem utilizar o modelo centralizado. O modelo de simulação distribuída suporta uma grande quantidade de dispositivos espalhados em múltiplos *host* em uma rede (e.g. LAN e WAN). Cada simulação está organizada em uma arquitetura de 3 camadas, como mostrado na Figura 3.

3.4. Algoritmos e Protocolos

No simulador proposto, novos algoritmos e protocolos podem ser criados através da classe *ApplicationComponent*. Os algoritmos são divididos em três métodos do Twisted:

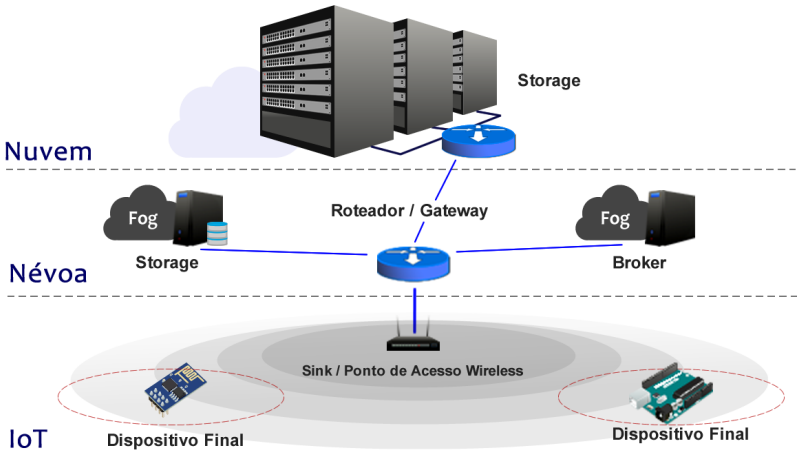

Figure 3. arquitetura de simulações do IoTFogSim.

- ***connectionMade()*** - Método onde usuário configura o comportamento de um nó durante a simulação ao receber uma conexão de entrada ou conectar-se a outro nó.
- ***dataReceived()*** - Método onde o usuário manipula os dados recebidos por um nó através do parâmetro *data* que guarda pacotes recebidos pela camada de rede.
- ***connectionLost()*** - Esse método permite ao usuário decidir o que acontece quando um nó perde a conexão de rede durante a simulação.

4. Avaliação de Desempenho

A fim de validar a proposta, o IoTFogSim foi submetido a uma avaliação de desempenho por meio da técnica de Planejamento de Experimentos, cujo objetivo é analisar o uso de memória e CPU pelos simuladores. Foi simulada uma aplicação MQTT em um computador com CPU 1.80GHz x 8, 64 bits, RAM 16Gb e HD 29,6 Gb.

Os experimentos basearam-se no Projeto Fatorial Completo [Jain 1991] técnica de análise de desempenho de sistemas computacionais, onde K fatores são observados alternando seus valores entre 2 níveis. Foram considerados 3 fatores A (Número de nós), (Número de eventos) B e C (Simulador). A Tabela 1 apresenta a matriz de experimentos. O simulador FogNetSim++ é a solução mais próxima da proposta e foi tomado como *baseline*.

Table 1. Projeto fatorial. Matriz de experimentos.

	Fatores		
EXP.	Número de nós (A)	Número de eventos (B)	Simulador (C)
1	10	1000	FogNetSim++
2	100	1000	FogNetSim++
3	10	10000	FogNetSim++
4	100	10000	FogNetSim++
5	10	1000	IoTFogSim
6	100	1000	IoTFogSim
7	10	10000	IoTFogSim
8	100	10000	IoTFogSim

5. Analise de resultados

Esta seção apresenta os resultados experimentos e uma comparação de performance entre os simuladores.

5.1. Uso de Memória

A análise da média de uso de memória leva a concluir que o simulador proposto utiliza menos memória do que o *baseline* em todos os cenários simulados. Por exemplo, no cenário de 100 nós e 10.000 eventos, que exige mais uso de memória, o uso médio de memória para proposta é de 35,8 Mb e para o FogNetSim++ utiliza 98,3 Mb. A Figura 4 mostra uma comparação do uso de memória realizado pelos simuladores.

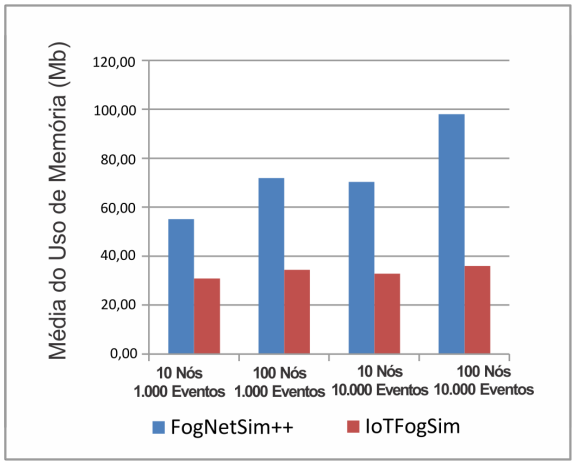

Figure 4. Comparação entre o uso de memória entre os simuladores.

5.2. Uso da CPU

Os resultados mostram que o uso de CPU pelo FogNetSim++ diminui quando os cenários mudam enquanto o uso da CPU pelo simulador proposto se mantém estatisticamente estabilizado e sem variação considerável. A Figura 5 mostra o gráfico do o uso da CPU. A média, de utilização da CPU pelo FogNetSim++ para os 10 nós e 1000 eventos é de 12% enquanto que o o uso máximo de CPU pelo IoTFogSim é de 2,33%, o que mostra que a proposta teve um desempenho melhor do que o *baseline*.

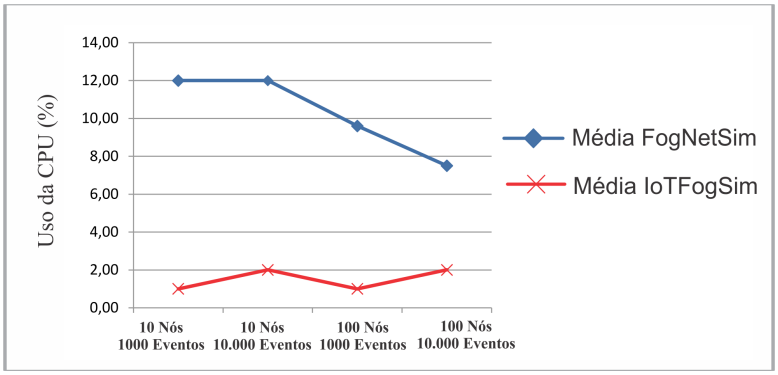

Figure 5. Comparação entre o uso de CPU entre os simuladores.

Os resultados mostram que o simulador proposto utiliza menos a memoria e a CPU do que o *baseline*, FogNetSim ++. Todavia é importante considerar que aplicações baseadas em python tendem a utilizar menos linhas e métodos do que aplicações desenvolvidas em java [Destefanis et al. 2016]. O IoTFogSim foi feito em python enquanto o *baseline* foi feito em java.

6. Conclusões e Trabalhos Futuros

Este artigo apresentou o IoTFogSim, um simulador de redes de computadores orientado a eventos voltado para IoT, computação em nuvem e em nevoa, baseado no conceito de simulações paralelas e distribuídas, o simulador proposto é capaz de simular um grande numero de dispositivos.

A avaliação de desempenho mostrou que a ferramenta proposta é capaz de exercer a função de um simulador de rede enquanto economiza o uso de memória e CPU. Por tanto, conclui-se que o simulador IoTFogSim pode auxiliar desenvolvedores de sistemas IoT, nuvem e nevoa no projeto e modelagem dos sistemas. Futuramente serão adicionados módulos de gráficos que permitam a visualização gráfica os dados produzidos.

References

Coutinho, A. A. T. R., Greve, F., and Prazeres, C. (2017). An Architecture for Fog Computing Emulation. *Wcga - Sbrc*, 15(1/2017).

D'Angelo, G., Ferretti, S., and Ghini, V. (2016). Simulation of the Internet of Things. *2016 International Conference on High Performance Computing and Simulation, HPCS 2016*, pages 1–8.

Destefanis, G., Ortu, M., Porru, S., Swift, S., and Marchesi, M. (2016). A statistical comparison of java and python software metric properties. In *2016 IEEE/ACM 7th International Workshop on Emerging Trends in Software Metrics (WETSoM)*, pages 22–28.

Gupta, H., Dastjerdi, A. V., Ghosh, S. K., and Buyya, R. (2016). ifogsim: A toolkit for modeling and simulation of resource management techniques in internet of things, edge and fog computing environments. *CoRR*, abs/1606.02007.

Han, S. N., Lee, G. M., Crespi, N., Heo, K., Van Luong, N., Brut, M., and Gatellier, P. (2014). DPWSim: A simulation toolkit for IoT applications using devices profile for web services. *2014 IEEE World Forum on Internet of Things, WF-IoT 2014*, pages 544–547.

Jain, R. (1991). *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Wiley Professional Computing. John Wiley & Sons Chichester.

Katal, A., Wazid, M., and Goudar, R. H. (2013). Big data: issues, challenges, tools and good practices. In *2013 Sixth international conference on contemporary computing (IC3)*, pages 404–409. IEEE.

Kertesz, A., Pflanzner, T., and Gyimothy, T. (2018). A Mobile IoT Device Simulator for IoT-Fog-Cloud Systems. *Journal of Grid Computing*, (June 2018):529–530.

Kraijak, S. and Tuwanut, P. (2015). A Survey on Iot Architectures , Protocols , Applications , Security , Privacy , Real-World. *11th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 2015)*, pages 1–6.

Labs and Twisted Matrix (2019). Twisted Documentation.

Naranjo, P. G. V., Shojafar, M., Vaca-Cardenas, L., Canali, C., Lancellotti, R., and Baccarelli, E. (2016). Big Data Over SmartGrid-A Fog Computing Perspective. *Proceedings of the 24th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2016)*, pages 22–24.

Patel, K., Vyas, S., Pandya, V., and saiyed, A. (2019). Iot: Leading challenges, issues and explication using latest technologies. In *2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)*, pages 757–762.

Peixoto, M., Genez, T., and Bittencourt, L. F. (2021). Hierarchical scheduling mechanisms in multi-level fog computing. *IEEE Transactions on Services Computing*, pages 1–1.

Perumalla, K. S. (2006). Parallel and distributed simulation: Traditional techniques and recent advances. In *Proceedings of the 2006 Winter Simulation Conference*, pages 84–95.

Pflanzner, T., Kertesz, A., Spinnewyn, B., and Latre, S. (2016). MobIoTSim: Towards a mobile IoT device simulator. *Proceedings - 2016 4th International Conference on Future Internet of Things and Cloud Workshops, W-FiCloud 2016*, pages 21–27.

Puliafito, C., Gonçalves, D. M., Lopes, M. M., Martins, L. L., Madeira, E., Mingozzi, E., Rana, O., and Bittencourt, L. F. (2020). Mobfogsim: Simulation of mobility and migration for fog computing. *Simulation Modelling Practice and Theory*, 101:102062. Modeling and Simulation of Fog Computing.

Qayyum, T., Malik, A. W., Khattak, M. A., Khalid, O., and Khan, S. U. (2018). FogNetSim++: A Toolkit for Modeling and Simulation of Distributed Fog Environment. *IEEE Access*, 6:63570–63583.

Ravi Teja, P. V., Chatterjee, S., Das, S. N., and Misra, S. (2015). Two-level mapping to mitigate congestion in machine to machine (M2M) cloud. *Proceedings - International Conference on 2015 Applications and Innovations in Mobile Computing, AIMoC 2015*, pages 103–108.

Shah, T. and Venkatesan, S. (2018). Authentication of iot device and iot server using secure vaults. In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 819–824.

Shipman, J. W. (2013). Tkinter 8.5 reference: a GUI for Python. *Computer*, pages 1–118.

Zeng, X., Garg, S. K., Strazdins, P., Jayaraman, P. P., Georgakopoulos, D., and Ranjan, R. (2017). IOTSim: A simulator for analysing IoT applications. *Journal of Systems Architecture*, 72:93–107.