UNIVERSIDADE ESTADUAL DE FEIRA DE SANTANA
Autorizada pelo Decreto Federal nº 77.496 de 27/04/76
Recredenciamento pelo Decreto nº17.228 de 25/11/2016
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
COORDENAÇÃO DE INICIAÇÃO CIENTÍFICA

XXIV SEMINÁRIO DE INICIAÇÃO CIENTÍFICA DA UEFS
SEMANA NACIONAL DE CIÊNCIA E TECNOLOGIA - 2020

COMUNICAÇÃO ENTRE INSTÂNCIAS DE NÉVOA EM AMBIENTE DE EMULAÇÃO MININET

Emerson Santos Lima[1] e Antonio Augusto Teixeira Ribeiro Coutinho[2]
Bolsista PROBIC, Graduando em Engenharia de Computação, Universidade Estadual de Feira de Santana, e-mail: elima@ecomp.uefs.br
2. Orientador, Departamento de Tecnologia (DTEC), Universidade Estadual de Feira de Santana, e-mail: acoutinho@uefs.br

PALAVRAS-CHAVE: virtualização; emulação; computação em névoa.

INTRODUÇÃO

As infraestruturas convencionais para oferta de serviços na Internet, baseadas em computação em nuvem (*Cloud Computing*) (NIST, 2011), estão se esforçando para atender o crescimento no número de dispositivos e do volume de dados gerados pela internet das coisas (*Internet of Things*, IoT) (GUBBI, 2013), ao mesmo tempo que procuram fornecer a qualidade de serviço esperada pelos clientes. Iniciativas recentes vêm propondo trazer os recursos de computação para a borda da rede, perto dos usuários, buscando assim estender benefícios da nuvem e cumprir requisitos de novas aplicações.

Neste sentido, a Computação em Névoa (*Fog Computing*) (BONOMI, 2012) é um modelo hierárquico em camadas, que busca oferecer um conjunto compartilhado e escalável de recursos de computação. Estes recursos são localizados entre a borda da rede e o *datacenter* de forma a preservar e estender os benefícios da nuvem através de serviços amplamente distribuídos em componentes da infraestrutura de rede. Dessa forma, além de minimizar o tempo de solicitação-resposta das aplicações, oferece facilidades como o reconhecimento de localização, distribuição geográfica, suporte à mobilidade e a redução na quantidade de dados transferidos à nuvem pelo processando local das informações.

Entretanto, o cenário em névoa é mais complexo para desenvolver, testar e gerenciar aplicações. Novos mecanismos e *frameworks* precisam ser projetados especificamente em névoa, onde a criação de bancos de teste (*testbed*) reais para prototipagem requerem a implantação de uma plataforma geograficamente distribuída, com diferentes equipamentos e links de comunicação, acarretando altos custos (ABREU et al., 2020).

Ambientes de simulação ou emulação representam uma alternativa viável e de baixo custo para avaliar soluções (COUTINHO, 2018). Embora sejam usados na modelagem de sistemas de nuvem, ainda estão em um estágio inicial em névoa (ABREU et al., 2020). O objetivo desta pesquisa é implementar a comunicação entre instâncias de névoa, emuladas em ambiente virtuais e executadas remotamente em diferentes máquinas *host* em uma rede local. A estrutura proposta permite a implantação e teste de componentes de névoa de forma escalável, onde seu design é compatível com tecnologias do mundo real, oferecendo suporte a sistemas de terceiros por meio de interfaces padronizadas.

MATERIAL E MÉTODOS

Para atingir o objetivo proposto, o primeiro passo foi um levantamento sobre modelos e padrões de IoT que podem ser aplicados para comunicação entre instâncias de névoa. Também, um estudo sobre a API do *framework* Fogbed (COUTINHO, 2018), uma solução baseada no ambiente de emulação de rede Mininet (LANTZ, 2010). Sua interface de configuração permite a distribuição de instâncias de névoa de forma escalável em diferentes máquinas (*hosts*), permitindo o suporte a serviços e protocolos do mundo real.

Após estudar os modelos de comunicação em IoT, foi adotada uma arquitetura baseada em *gateways* de baixo custo como nós de névoa. Esses *gateways* possuem recursos de processamento e memória limitados, porém são capazes de executar serviços e protocolos de comunicação necessários para atuar como controladores, concentrando as funções de acesso à dispositivos sensores e atuadores próximos, na mesma rede local.

Em seguida, foram levantados requisitos funcionais dos módulos, além da adaptação dos componentes necessários para que o *gateway* IoT fosse capaz de disponibilizar serviços de comunicação e acesso aos dispositivos na arquitetura empregada. Neste processo, foi definido um protocolo de comunicação através e entre as instâncias de névoa baseado em tecnologias bem conhecidas e amplamente utilizadas na *web* das coisas (*Web of Things*, WoT) (GUINARD, 2011). Em nossa proposta, os nós de névoa foram implementados virtualmente executando o núcleo (*kernel*) do sistema operacional Linux, uma máquina virtual Java[1], uma implementação da especificação OSGi[2], e um servidor MQTT[3] (*Message Queuing Telemetry Transport*). Essa configuração foi implementada através da interface de contêineres Docker[4], onde uma única imagem do sistema pode ser usada para instanciar todos os nós em um ambiente virtual.

Por fim, um experimento foi realizado para avaliar os limites de desempenho dos nós virtuais de névoa propostos no controle e gerenciamento dos dispositivos da IoT em situação extrema de utilização. Através da API de topologia do Fogbed foram implementados *scripts* de configuração dinâmica para rede emulada entre as instâncias de névoa virtuais. Na realização dos experimentos, foram utilizadas uma rede de computadores física e estações de trabalho do Laboratório de Redes e Sistemas Distribuídos (LARSID) da Universidade Estadual de Feira de Santana (UEFS), além de recursos do bolsista e do orientador na metade final do trabalho, devido ao isolamento do laboratório pela pandemia de COVID-19 no Estado da Bahia.

ANÁLISE E DISCUSSÃO DOS RESULTADOS

Em nossa proposta, o *gateway* usa um *middleware* orientado à serviços (*Enterprise Service Bus*, ESB) que mantém um container de serviços, com objetivo de prover funcionalidades para acesso e controle dos dispositivos físicos da IoT como, por exemplo, obter a temperatura de uma sala, ligar ou desligar uma lâmpada, abrir uma porta, etc.

O *middleware* ESB fornece acesso ao *gateway* através de interfaces baseadas em serviços *web* REST (*Representational State Transfer*) (FIELDING e TAYLOR, 2000)

[1] https://www.oracle.com/java/
[2] http://www.osgi.org/
[3] https://mqtt.org/
[4] https://www.docker.com/

para os clientes da aplicação, além de oferecer uma camada de gerenciamento, configuração e monitoramento dos serviços e mensagens. Outro componente desta arquitetura é o *broker*, que emprega um servidor MQTT para troca de mensagens entre os *gateway*s, e entre os *gateway*s e os dispositivos IoT. O MQTT usa o TCP/IP como protocolo de transporte, oferecendo mecanismos de notificação de desconexão no modelo cliente/servidor seguindo o padrão *publish-subscribe*.

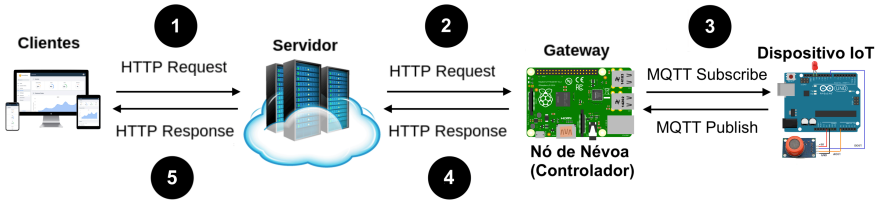

Figura 1. Fluxo de mensagens na requisição de dados por cliente a um dispositivo IoT.

Os nós de névoa atuam na borda da rede recebendo e interpretando requisições RESTful via HTTP (*Hypertext Transfer Protocol*). A Figura 1 mostra os passos necessários para acessar dados de dispositivos físicos disponíveis através da arquitetura em névoa proposta: (1) a partir da interação do usuário com a aplicação, o cliente envia requisições HTTP para um servidor *web*, com o objetivo de obter dados de um dispositivo físico; (2) ao receber a requisição, o servidor *web* na nuvem direciona o pedido para o *gateway* responsável pelo dispositivo; (3) o gateway se comunica com o dispositivo em sua rede usando um *broker* MQTT para obter e enviar dados encapsulados nas respostas (4 e 5) do protocolo HTTP ao cliente.

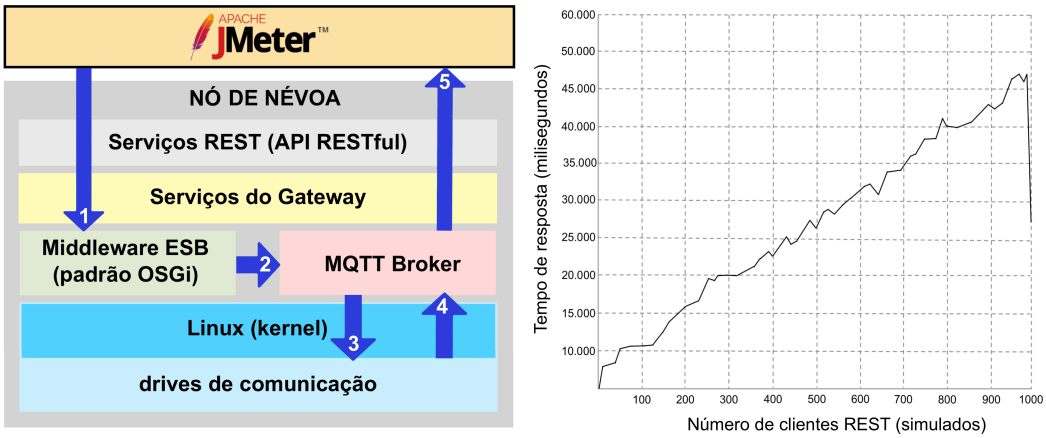

Figura 1. (a) Visão da arquitetura em camadas implementada em nós de névoa. (b) Tempo de resposta nó de névoa emulado x número de clientes simultâneos.

A Figura 2(a) ilustra o modelo em camadas do *gateway* e as tecnologias usadas em sua implementação, onde as setas numeradas representam o fluxo de mensagens entre os componentes. O Apache JMeter[5] não faz parte do nó de névoa, assumindo o papel da aplicação cliente no experimento, onde os passos 1→2→3→4→5 são medidos para avaliar o desempenho da arquitetura proposta. Usando o Apache JMeter, foi possível simular um número n de clientes, através de requisições RESTful simultâneas ao nó de névoa virtual.

[5] https://jmeter.apache.org/

O cenário emulado no Mininet está representado na Figura 1. A máquina utilizada na emulação dos componentes e execução dos testes possui 8GB de memória RAM DDR3, processador Intel i5 de 3Ghz, e sistema operacional Linux Ubuntu 14.04 LTS.

A Figura 2(b) mostra a influência que a arquitetura e número de clientes (requisições simultâneas) exercem sobre o desempenho (tempo de resposta). É possível notar que o tempo de resposta cresce em função do número de clientes, dada a limitação de processamento do *gateway*. Ao aproximar-se de mil clientes, nota-se a ocorrência de *timeout*, resultado do limite no tempo de resposta estabelecido na aplicação. No gráfico, a queda abrupta no tempo de resposta representa o momento de colapso do ESB.

CONSIDERAÇÕES FINAIS

O experimento apresentado faz parte de um trabalho que busca avaliar a escalabilidade na emulação de ambientes IoT em névoa para responder questões como: "quantos nós de névoa IoT podem ser emulados em uma máquina *host* no Mininet?". Para isso, foi necessário estudar e implementar a comunicação necessária entre os nós de névoa usando tecnologias e interfaces padronizadas do mundo real. Infelizmente, em virtude do isolamento dos laboratórios da UEFS pela COVID-19, não foi possível finalizar todos os experimentos, onde o objetivo é avaliar a escalabilidade do ambiente virtual executado em diferentes máquinas *host* em uma rede local. Entretanto, o trabalho remoto para implementação dos experimentos foi realizado, e esperamos concluir os trabalhos assim que as atividades normais forem retomadas pela comunidade universitária.

REFERÊNCIAS

FIELDING, R. T., TAYLOR, R. N., 2000. Architectural styles and the design of network-based software architectures. Irvine: University of California, Irvine. vol. 7.

LANTZ, B et al. 2010. A network in a laptop: rapid prototyping for software-defined networks. In: Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks. pp. 1-6.

GUINARD, D. et al. 2011. From the internet of things to the web of things: Resource-oriented architecture and best practices. In: Architecting the Internet of things. Springer, Berlin, Heidelberg. p. 97-129.

NIST. 2011. The NIST Definition of Cloud Computing. Disponível em: http://faculty.winthrop.edu/domanm/csci411/Handouts/NIST.pdf. Acesso em: 26 jul. 2019.

BONOMI, F. et al. 2012. Fog computing and its role in the internet of things, Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing. ACM, pp. 13-16.

GUBBI, J. et al. 2013. Internet of Things (IoT): A vision, architectural elements, and future directions, Future generation computer systems 29.7: 1645-1660.

COUTINHO, A. et al. 2018. Fogbed: A rapid-prototyping emulation environment for fog computing, in Communications Workshops (ICC Workshops), 2018 IEEE International Conference, pp. 1-7.

ABREU, D. P. et al. 2020. A comparative analysis of simulators for the Cloud to Fog continuum. Simulation Modelling Practice and Theory, vol.101, pp.102029.