15ª ESCOLA REGIONAL DE REDES DE COMPUTADORES
25 – 29 de setembro de 2017
Santa Maria – RS

ANAIS

Editora
Sociedade Brasileira de Computação – SBC

Organização
Carlos Raniery Paula dos Santos

Execução
Universidade Federal de Santa Maria - UFSM

Editor
Carlos Raniery Paula dos Santos (UFSM)

Realizado por
Sociedade Brasileira de Computação – SBC

Capa: Carlos Raniery Paula dos Santos

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Escola Regional de Redes de Computadores (15.: 25 – 29 set 2017: Santa Maria - RS)
Anais/Organização : Carlos Raniery Paula dos Santos, 2017.

Conhecido também como ERRC 2017.
1. Redes de Computadores. 2. Sistemas Distribuídos.
I. ERRC (15.: 25–29 set 2017: Santa Maria - RS). II. Universidade Federal de Santa Maria. III. Santos, Carlos Raniery Paula dos

ERRC 2017

http://www.ufsm.br/errc2017

Comissão Organizadora

Coordenação Geral

•Prof. Carlos Raniery Paula dos Santos (UFSM)

Coordenação Local

•Profa. Marcia Pasin (UFSM)
•Profa. Roseclea Duarte Medina (UFSM)
•Profa. Simone Ceolin (CTISM- UFSM)

Organização do Comitê de Programa

•Prof. Érico Hoff Amaral (Unipampa)
•Prof. Gleizer Voss (IF Farroupilha/São Vicente)

Organização de caravanas

•Prof. Roben Lunardi (IFRS)

Organização de Minicursos, Palestras e Oficinas

•Profa. Simone Ceolin (CTISM- UFSM)
•Prof. Tiago Coelho Ferreto (PUC-RS)

Organização de Divulgação e Patrocínios

•Profa. Márcia Henke (CTISM- UFSM)
•Prof. Marcelo da Silva Conterato (SENAC-RS)

Webmaster

•Thales Nicolai Tavares (UFSM)

Organização Local

- Anderson Monteiro (IFFar/UFSM)
- Brenda Salenave (CT - UFSM)
- Caroline Chagas (CT - UFSM)
- Daniel Matheus (CT - UFSM)
- Dênes Vargas (CT - UFSM)
- Filipe Simões (CT - UFSM)
- Gabriel Cardoso (CT - UFSM)
- Ivana Fischer (CTISM - UFSM)
- Jhillian Bianchi (CT - UFSM)
- Larissa Souto (CTISM - UFSM)
- Leonardo Marcuzzo (UFSM)
- Luísa Perin Lucca (CT - UFSM)
- Maurício Matter Donato (CT - UFSM)
- Nilton Camargo (UFSM)
- Otávio Prestes (CTISM - UFSM)
- Paulo Vinicius Cardoso (CT - UFSM)
- Rafael Guimarães (CT - UFSM)
- Thales Nicolai Tavares (UFSM)
- Vinícius Fülber Garcia (UFSM)
- Yagor Duarte (CTISM - UFSM)

ERRC 2017

http://www.ufsm.br/errc2017

Comitê técnico do programa

•Adenauer Yamin (UCPEL e UFPEL)
•Alberto Schaeffer-Filho (UFRGS)
•Alex Camargo (Unipampa)
•Aliane Krassmann (IFFar)
•André Peres (IFRS - Campus Porto Alegre)
•Andre Du Bois (UFPEL)
•Andrea Charao (UFSM)
•Andrea Krob (Unilasalle)
•Carlos Vinícius Rasch Alves (FATEC SENAC Pelotas)
•Cesar Loureiro (IFRS)
•Claudio Schepke (Unipampa)
•Cristiano Both (UFCSPA)
•Cristiano Costa (Unisinos)
•Cristina Nunes (PUC RS)
•Dalvan Griebler (PUC RS)
•Diego Kreutz (University of Luxembourg)
•Eduardo Back (Faculdade IENH)
•Eduardo Monks (FATEC SENAC Pelotas)
•Ewerton Salvador (UNIPÊ)
•Fabrício Herpich (UFRGS)
•Fábio Rossi (IFFar)
•Felipe Becker Nunes (UFRGS)
•Fernando Prass (UNIFRA e ULBRA)
•Flávio Santos (Chaordic Systems)
•Gerson Soares (Università degli Studi di Milano)
•Glederson Santos (UFSul)
•Gleizer Voss (IFFar)
•Guilherme da Cunha Rodrigues (IFSul)

•Henry Carvalho (EMBRAPA)
•José Jair Santanna (University of Twente)
•Juliano Wickboldt (UFRGS)
•Leandro Bertholdo (UFRGS)
•Leonardo Bays (UFRGS)
•Leonardo Pinho (Unipampa)
•Lisandro Zambenedetti Granville (UFRGS)
•Lucas Bondan (UFRGS)
•Luciano Ignaczak (Unisinos)
•Lucio Prade (Unisinos)
•Luis Knob (PUC RS)
•Maicon Kist (UFRGS)
•Marcelo Caggiani Luizelli (UFRGS)
•Marcelo Conterato (PUC RS)
•Marcia Henke (UFSM)
•Marcia Pasin (UFSM)
•Marcio Pohlmann (FATEC SENAC POA e UFRGS)
•Marco Trentin (UPF)
•Marinho Barcellos (UFRGS)
•Mauricio Pilla (UFPEL)
•Miguel Baggio (IFSul)
•Oscar Caicedo (UFRGS)
•Rafael Bastos (UFPEL)
•Rafael Esteves (IFRS)
•Rafael Kunst (UFRGS)
•Raul Ceretta Nunes (UFSM)
•Regio Michelin (IFRS)
•Renata Reiser (UFPEL)
•Ricardo Becker (UCS)
•Ricardo Schmidt (SIDN Labs e University of Twente)
•Ricardo Luis dos Santos (UFRGS)
•Roben Lunardi (IFRS)
•Rodolfo Antunes (Unisinos)
•Rodrigo Calheiros (Western Sydney University)
•Rodrigo Righi (Unisinos)
•Rogério Turchetti (UFSM)
•Simone Ceolin (UFSM)

- Taisy Weber (UFRGS)
- Tiago Ferreto (PUC RS)
- Tiago Antônio Rizzetti (UFSM)
- Vinícius Guimarães (IFSul - Campus Charqueadas)
- Walter Priesnitz Filho (UFSM)
- Weverton Cordeiro (UFRGS)

Apresentação

É um prazer e uma distinção organizar um evento de tamanha relevância para a computação no estado, mais ainda por completar-se 10 anos desde a primeira vez que Santa Maria teve o privilégio de sediá-lo. Agradecemos à comunidade pela confiança em contribuir com uma das escolas regionais mais tradicionais e relevantes nacionalmente. No contexto desafiador em que o país se encontra nesse ano de 2017, é ainda mais importante reforçar o papel primordial do ERRC como fórum de discussão, encontros e divulgação dos melhores trabalhos produzidos pela comunidade de pesquisa regional.

A programação do ERRC 2017 está diversificada, abrangente e tem excelente qualidade; nesse sentido, a contribuição da comunidade foi fundamental para a valorização do evento e o fortalecimento da Ciência e Tecnologia no estado. A programação engloba 6 sessões técnicas com apresentação de 22 artigos científicos completos, selecionados por meio de um rigoroso trabalho de revisão, 5 palestras proferidas por pesquisadores internacionalmente renomados. São oferecidos ainda 5 minicursos, voltados à formação e atualização dos participantes em temas de ponta.

A excelência das atividades programadas nesta edição é reflexo da competência e empenho dos seus respectivos coordenadores. Um agradecimento muito especial a Marcia Pasin (UFSM), Roseclea Duarte Medina (UFSM), Simone Ceolin (CTISM-UFSM), Raul Ceretta Nunes (UFSM), Érico Hoff do Amaral (Unipampa), Gleizer Voss (IF-Farroupilha), Tiago Coelho Ferreto (PUC-RS), Roben Lunardi (IFRS), Márcia Henke (CTISM- UFSM) e Marcelo da Silva Conterato (SENAC-RS). Exaltamos ainda o trabalho voluntário, intenso, atencioso e contínuo, realizado pelos colegas do Comitê de Organização Local.

Agradecemos aos integrantes do Comitê Consultivo do ERRC, pelos aconselhamentos prestados à organização do ERRC 2017. Gostaríamos ainda de agradecer aos patrocinadores do simpósio: aos órgãos de fomento, CAPES e FAPERGS, aos nossos apoiadores e às empresas patrocinadoras por valorizarem e reconhecerem o ERRC como um evento importante para o fomento à pesquisa e inovação.

Nosso agradecimento especial às nossas instituições, especialmente, ao Departamento de Computação Aplicada (DCOM) e ao Departamento de Linguagens e Sistemas de Computação

(DLSC), ao Colégio Técnico Industrial de Santa Maria (CTISM), ao Centro de Tecnologia e à Reitoria da UFSM, pelo indispensável suporte para a realização do ERRC.

Em nome do Comitê Organizador do ERRC 2017, desejamos a todos uma semana agradável em Santa Maria, rica em discussões e encontros.

Carlos Raniery Paula dos Santos

Coordenador Geral da ERRC 2017

Sumário

I

ERRC - Sessão de Técnica

Análise e comparação de técnicas e aceleradores de processamento de pacotes

Leonardo da C. Marcuzzo[1], Carlos R. P. dos Santos[1]

[1]Departamento de Computação Aplicada - Universidade Federal de Santa Maria (UFSM)
Avenida Roraima nº 1000 - 91.105-900 - Santa Maria - RS - Brasil

{lmarcuzzo,csantos}@inf.ufsm.br

Abstract. *The major advances in network interfaces of servers were not accompanied by the network stack of traditional operating systems. While Network Interface Cards (NICs) are able to achieve up to 100 Gbit/s of traffic, operating systems can achieve only 1/10 of this performance. Due to this, techniques for faster packet processing are being developed in order to reduce performance loss. This paper presents some of these techniques and frameworks where these are implemented, and shows a performance comparison of those frameworks against traditional operating systems, to justify their usage.*

Resumo. *Os grandes avanços no hardware de interfaces de rede de servidores comuns não foram acompanhados pela pilha de rede de sistemas operacionais tradicionais. Enquanto interfaces de rede são capazes de atingir até 100 Gbits/s de tráfego, sistemas operacionais não conseguem processar 1/10 desta quantidade de tráfego. Desta forma, algumas técnicas para acelerar o processamento de pacotes estão sendo desenvolvidas com o objetivo de mitigar esta perda de desempenho. Este artigo apresenta algumas técnicas, bem como frameworks que as implementam, e realiza uma comparação com sistemas operacionais tradicionais, justificando o uso destes aceleradores.*

1. Introdução

A demanda crescente pela capacidade de roteamento e processamento de pacotes, devido a universalização do acesso à Internet, ao uso de cada vez mais conteúdo midiático e pela disponibilidade de conexões mais rápidas requer tecnologias capazes de suportar este crescimento [ITU 2016]. Em *datacenters* e *Points-of-Presence* (PoPs), esta demanda é suprida através do uso de equipamentos proprietários especialmente desenvolvidos para o encaminhamento e processamento de pacotes (*i.e., middleboxes*).

No entanto, a utilização destes equipamentos resulta em um ecossistema inflexível, dominado por tecnologias proprietárias, com altos custos de compra e manutenção, o que impede a entrada de empresas menores no setor [Sherry et al. 2012]. Uma das alternativas ao uso de *middleboxes* é a utilização de servidores genéricos (*Commercial off-the-self* - CoTS) para executarem essas funcionalidades, aproveitando-se da disponibilidade de interfaces de rede capazes de enviar e receber uma quantidade de tráfego comparável a *middleboxes*. No entanto, a pilha de rede de sistemas operacionais tradicionais não é capaz de processar esta quantidade de dados, mesmo em arquiteturas *multi-core*, por serem desenvolvidos com foco em generalidade e compatibilidade, ao invés de alto desempenho. Como exemplo, testes realizados em [Salopek et al. 2014]

mostram que a vazão da pilha de rede do Linux é inferior a 10Gbit/s, mesmo utilizando a *New API* [Salim 2005].

Considerando esta defasagem dos sistemas operacionais, e impulsionados por novas tecnologias de rede (*e.g.*, Redes Definidas por Software - SDN e Virtualização de Funções de Rede - NFV) que podem se beneficiar destes sistemas, técnicas focadas na melhoria de desempenho no encaminhamento e processamento de pacotes em sistemas tradicionais estão sendo desenvolvidas e implementadas através de *frameworks*, com o objetivo de tornar servidores CoTS uma alternativa de menor custo com relação as soluções proprietárias existentes [García-Dorado et al. 2013]. Estes *frameworks* de aceleração de pacotes são capazes de contornar ou substituir a pilha de rede de sistemas tradicionais e utilizar essas técnicas para que aplicações no *userspace* possam ter um desempenho comparável ao de *middleboxes*, permitindo que funcionalidades exclusivas desses possam, enfim, serem implementadas em sistemas tradicionais.

Dado os benefícios que podem ser obtidos ao utilizar aceleração de pacotes, este artigo tem como objetivo apresentar uma visão geral sobre algumas das técnicas mais utilizadas, como processamento em lotes, pré-alocação de recursos e *kernel bypass*. Dois *frameworks* onde essas são implementadas, DPDK [Intel 2014] e Netmap [Rizzo 2012], terão seu desempenho avaliado com relação a vazão, latência e utilização de recursos e comparados com a pilha de rede tradicional do Linux, com o objetivo de demonstrar sua eficiência em um cenário com alto consumo de recursos de rede.

O restante do artigo está organizado da seguinte forma: A seção dois apresenta uma visão detalhada das técnicas e *frameworks* que as implementam. Após, na seção três, é apresentada a metodologia de testes e os resultados obtidos, bem como uma discussão sobre esses. Por fim, na seção quatro são apresentadas as conclusões dos autores.

2. Processamento Rápido de Pacotes

Até a versão 2.6 do *kernel* do Linux, a pilha de rede funcionava primariamente por interrupções. Desta forma, toda vez que um pacote é recebido pela interface de rede, ele é colocado em uma fila circular (*i.e., rings*), e chama uma interrupção. Após, o pacote é copiado desta fila para uma área de memória interna, conhecida como *packet kernel buffer*, que copia novamente para a pilha de rede, onde é processado, para que só assim possa ser disponibilizado para o *userspace* (através de outra cópia). A implementação da *New API* otimizou alguns destes processos para casos onde há uma grande quantidade de tráfego, mitigando interrupções (após a primeira interrupção, é feito um *pooling* para os pacotes seguintes) e, quando o sistema está sobrecarregado, permitindo que pacotes sejam descartados ainda na interface de rede, evitando cópias desnecessárias.

No entanto, ainda existem diversas limitações nessa nova implementação, como por exemplo a alocação e desalocação de recursos para cada pacote, o acesso serial, cópias múltiplas entre o *kernel* e aplicações no *userspace* e trocas de contexto. Em suma, é possível dizer que, embora a *New API* seja um avanço com relação ao estado anterior da pilha de rede, ele ainda não é adequado para ser usado em ambientes com grande quantidade de pacotes a serem processados.

Dado estas limitações, várias técnicas foram propostas com o objetivo de otimizar o desempenho dos sistemas operacionais quando trabalham com grande quantidade de dados.

2.1. Técnicas de aceleração de pacotes

A seguir, algumas das técnicas mais utilizadas serão apresentadas e detalhadas, apresentando seus benefícios e deficiências [Hermsmeyer et al. 2009]. Em [Tsiamoura et al. 2014], algumas destas técnicas foram analisadas com o objetivo de mostrar sua utilização no contexto de utilização com roteadores virtuais. Neste artigo, o foco da análise foi sobre encaminhamento e processamento de pacotes de máquinas virtuais.

- **Pré-alocação e reuso de recursos:** Tipicamente os recursos para o recebimento e processamento de pacotes são alocados na hora em que o pacote é recebido (ou enviado), e liberados após o envio para a próxima camada. A utilização de um espaço de memória alocado em um momento prévio (*e.g.*, na inicialização do *driver*) para o armazenamento de pacotes, e o reuso desta memória para armazenar os pacotes seguintes eliminam o gasto de processamento causado por esses métodos. A desvantagem é que isto resulta em um consumo maior de memória, já que mesmo que nenhum pacote esteja sendo processado ou encaminhado, a memória deve se manter alocada.
- **Suporte a múltiplos rings:** Interfaces de rede modernas possuem múltiplos *rings* de recebimento e envio, que podem ser utilizados de forma independente. Isto permite que a carga seja balanceada em sistemas *multi-core*, com aplicações, núcleos ou canais de memória diferentes escrevendo em *rings* de forma paralela. No entanto, o uso de múltiplos núcleos para recebimento e envio de pacotes reduz os recursos disponíveis para outras aplicações, além de pacotes poderem chegar fora de ordem, ocasionando problemas de ordenação em algumas aplicações.
- **Processamento em lotes:** Para reduzir a quantidade de cópias entre a interface de rede e o restante do sistema, os pacotes podem ser copiados em conjunto. Esta técnica é bastante efetiva por reduzir a quantidade de acessos ao *hardware* da interface de rede e a quantidade de cópias do *kernel* ou *framework* para a aplicação do usuário. Porém, isto causa um aumento no *jitter* e latência nos pacotes, pois estes devem esperar o processamento de todos os pacotes, ou um *timer* expirar, caso não haja pacotes suficientes no lote.
- **Kernel bypass:** Finalmente, a pilha de rede pode ser inteiramente ignorada pela interface de rede, que pode entregar diretamente os pacotes para a aplicação do usuário, criando um *fastpath* entre os descritores da placa e a aplicação. No entanto, para que esta técnica funcione, a aplicação deve possuir *drivers* próprios e ser capaz de processar os pacotes como vieram da interface (*e.g.*, desencapsular o Frame).

2.2. Frameworks de aceleração de pacotes

As técnicas citadas anteriormente podem funcionar em conjunto para que um desempenho melhor possa ser obtido. Assim, *frameworks* de aceleração de pacotes apresentam várias técnicas implementadas em diferentes camadas do sistema (*e.g., drivers, kernel, userspace*), sendo que todos estes *frameworks* possuem como foco a otimização de aplicações no *userspace*, por ser o ambiente onde a grande maioria das aplicações são implementadas, e podem ser otimizadas de forma segura. Em alguns casos, estes *frameworks* podem substituir inteiramente a pilha de rede, desde que as aplicações possam tratar os pacotes da forma que chegam da interface física.

Assim, foram identificados três *frameworks* de código aberto popularmente utilizados na literatura, e embora utilizem técnicas semelhantes, possuem arquiteturas diferentes. Nota-se que existem esforços de grandes empresas (e.g., Cisco, Solarflare) neste mesmo sentido, porém são soluções proprietárias.

- **Packet_mmap:** [Linux 2005]: Desenvolvido como uma forma mais efetiva de monitorar o tráfego de interfaces de rede no *kernel* do Linux. É implementado através de modificações nos *sockets* para que os *buffers* dos pacotes sejam alocados em uma área de memória compartilhada entre o *kernel* e o *userspace*, reduzindo o número de cópias necessárias. No entanto, os pacotes ainda precisam ser processados pela pilha de rede do sistema, o que impede que outras técnicas para a aceleração de pacotes possam ser utilizadas em conjunto.
- **netmap** [Rizzo 2012]: O *netmap* é um *framework* desenvolvido com o objetivo de reduzir o consumo de recursos necessários para encaminhar tráfego entre o *hardware* e o *userspace*. Interfaces de rede associadas com o *netmap* são controladas por um *driver* próprio, enquanto que a pilha de rede do *kernel* pode funcionar em paralelo com outras interfaces, perdendo o acesso às controladas pelo *netmap*. Das técnicas apresentadas anteriormente, o *netmap* utiliza *kernel bypass, batching* e suporte a multi-queues (se disponível no hardware), além de possuir proteções com relação a programação direta do *hardware* e do uso da memória (que precisam ser validados pelo sistema). Recentemente, foi adicionado suporte à criação de *pipes* internos (e.g, para comunicação entre Virtual Machines (VMs) no mesmo host). No entanto, sua alocação de *buffers* não é feita de forma dinâmica, o que impede que *zero-copy* seja usado de maneira efetiva. Sua implementação possui também um comutador virtual próprio (VALE) e, segundo os autores, é capaz de atingir 14.88 milhões de pacotes (i.e., a maior quantidade possível de pacotes de 64B em uma interface de 10GbE) utilizando apenas um único núcleo.
- **DPDK** [Intel 2014]: Desenvolvido pela Intel, o Data Plane Development Kit (DPDK) é um *framework* comparável ao *netmap* que utiliza técnicas semelhantes, além de possuir alocação dinâmica de *buffers* e mais funcionalidades no *userspace*. Da mesma forma que o *netmap*, o DPDK controla diretamente os descritores das interfaces, permitindo que aplicações do usuário se comuniquem diretamente com o *hardware*, diminuindo o *overhead*. Um dos diferenciais do DPDK com relação ao *netmap* é um suporte melhor para a manipulação de pacotes em arquiteturas *multi-core*, sendo capaz de distribuir de maneira otimizada o processamento nos canais de memória RAM. Como exemplo, é possível configurar o DPDK para utilizar um núcleo inteiramente para o *pooling* e encaminhamento de pacotes , deixando os outros para realizar outras funções (modo *pipeline*), ou então dividir igualmente a carga entre eles (modo *run-to-completion*).

A Figura 1 apresenta a arquitetura de alto nível destes *frameworks*, de modo a mostrar diferenças entre suas arquiteturas. É possível ver que o DPDK roda completamente no *userspace*, utilizando uma *thread* para fazer o encaminhamento dos pacotes. VMs e aplicações são conectadas a esta *thread*, que por sua vez encaminha os pacotes para o seu destino de maneira eficiente. Já o *netmap* é implementado como um módulo do *kernel* que disponibiliza uma área de memória compartilhada com o *userspace*, criando um *fastpath*.

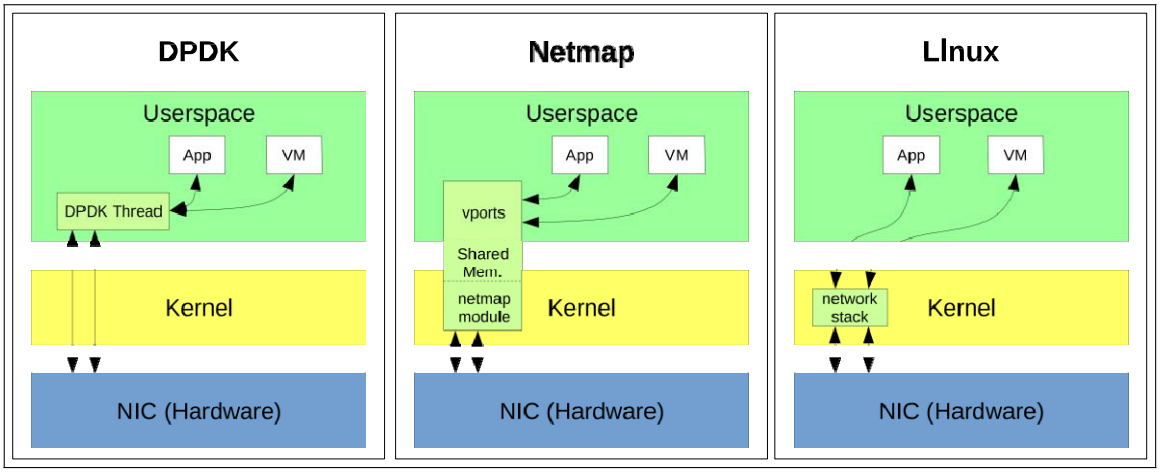


Figura 1. Comparação entre a arquitetura dos sistemas testados

Neste caso, cada VM ou aplicação têm direito a uma parte desta área onde realiza escrita e leitura. O módulo por sua vez é responsável por gerenciar e encaminhar os pacotes nesta área, fazendo cópias em lote entre áreas diferentes. Por fim, na pilha de rede tradicional do Linux, as interfaces das VMs são conectadas ao módulo *vhost*, que delega ao Linux o roteamento e encaminhamento de pacotes para ser processado pelo *kernel*. Pelo *Packet_mmap* apenas fazer alterações de baixo nível na arquitetura do Linux, sua arquitetura de alto nível é igual a do Linux.

3. Metodologia e Resultados

Dado os *frameworks* descritos anteriormente, optou-se pela criação de um cenário virtualizado, composto por duas máquinas virtuais, cada uma com uma interface de rede, um núcleo e 1Gb de RAM, executando o sistema operacional Debian 8 e virtualizadas através do *hypervisor* KVM (*Kernel-based Virtual Machine*). O *host* destas máquinas virtuais possui um processador Intel Core i5 2.2Ghz, com dois núcleos (quatro *threads*) e 8Gb de RAM DDR3@1066Mhz.

O cenário interconecta as duas VMs através do *host*, utilizando três métodos diferentes. O primeiro método é a interconexão das máquinas através de Linux Bridges. Tomou-se este como base por ser a forma padrão de conectar VMs entre si e entre a rede externa no Linux, além de utilizar a New API. Considerando que o packet_mmap é apenas uma extensão da pilha de rede do Linux, ele não foi considerado nos testes por já utilizar-se a pilha tradicional. O segundo método utiliza o *netmap* para a interconexão das VMs, o que foi feito através da inserção do módulo no *kernel* e a criação de interfaces paravirtualizadas do tipo `ptnetmap`, que por sua vez são conectadas ao comutador VALE disponibilizado pelo módulo. O terceiro método utiliza o exemplo `l2fwd` disponibilizado pelo DPDK para interconectar as VMs, definindo-se as interfaces das VMs com o tipo `dpdkvhostuser`.

Foram escolhidas as formas mais simples de interconexão nos três métodos, para que não houvesse disparidade no consumo de outros recursos do *host* e não foram realizadas nenhum tipo de otimizações nos métodos (*e.g.*, afinidade de cpu).

Assim, para a execução dos testes, foi instalado em ambas VMs a

aplicação NetPIPE [Turner and Chen 2002], para realizar testes de vazão e latência, métricas comumente utilizadas para desempenho e definidas na RFC 2544 [Bradner and McQuaid 1999]. A ferramenta foi escolhida por realizar testes com diferentes tamanhos de blocos, variando continuamente seu tamanho, de modo a simular um uso mais realista das conexões.

3.1. Resultados e Discussão

Após a realização dos testes e coleção das métricas, verificou-se que os dois *frameworks* se mostraram superiores a implementação da pilha tradicional do Linux, comprovando assim a efetividade das técnicas descritas anteriormente.

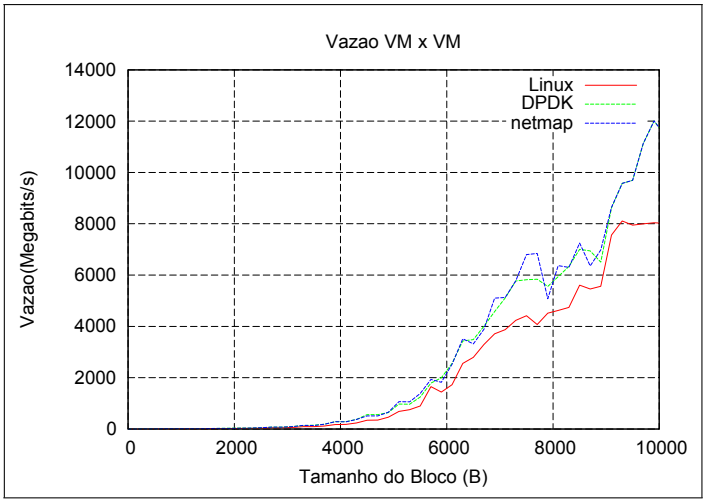


Figura 2. Teste de vazão entre duas VMs

A Figura 2 apresenta o resultado da vazão para o cenário utilizado. É possível ver que tanto o DPDK como o *netmap* possuem uma vazão muito parecida, sendo que ambas são superiores a implementação original do Linux. O desempenho de ambas foi de 12 Gbit/s para o tamanho máximo dos pacotes (10kB), enquanto que o Linux chegou a 8 Gbit/s. Mesmo que o *netmap* precise realizar cópias entre a memória, isto não afetou o seu desempenho em comparação com o DPDK.

Nota-se que o desempenho do Linux foi avaliado utilizando a tecnologia *vhost* (*i.e.*, algumas das operações de rede são movidas para o *kernel*, que já é capaz de reduzir latência, número de cópias e CPU em comparação a implementação original), mesmo assim ficando abaixo do desempenho dos frameworks.

Já a Figura 3 apresenta os resultados de latência para o cenário. Novamente, o desempenho de ambos os *frameworks* foi superior ao do Linux. No entanto, o desempenho do DPDK foi superior ao do *netmap*. Isto pode ser explicado pelo fato do DPDK executar inteiramente no *userspace* enquanto que o *netmap* precisa que seja realizada uma cópia para uma área de memória do kernel.

Tanto para vazão como para latência, o resultado dos *frameworks* foi superior ao do Linux. Verificou-se também que o consumo de CPU e memória, tanto das VMs quanto do host nunca chegou ao máximo, o que poderia impactar a validade dos resultados. Vale ressaltar que tanto o DPDK como o *netmap* necessitam que configurações adicionais sejam feitas no *host* e nas VMs, e o consumo de memória e CPU é maior do que o *vhost*.

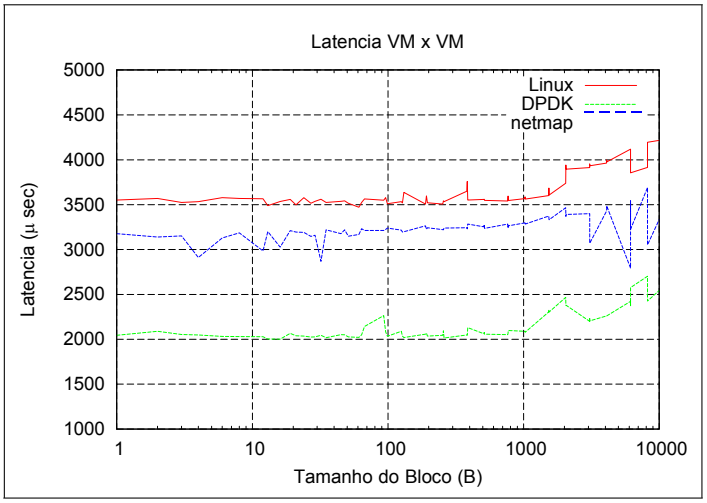


Figura 3. Teste de latência entre duas VMs

Os resultados de latência foram na ordem dos microssegundos, devido ao fato de todas máquinas e aplicações executarem no mesmo *host*. Mesmo assim, nota-se claramente um desempenho superior dos *frameworks*, principalmente do DPDK, que reduziu a latência quase pela metade quando comparado ao Linux. Acredita-se que isto se deve ao fato de este ser executado inteiramente no *userspace*, o que reduz trocas de contexto e outras operações que custam ciclos ao processador. O seu uso elevado de recursos, causado pela necessidade de *HugePages* e um núcleo inteiramente dedicado ao *pooling* devem ser considerados na hora da implementação. No caso dos testes realizados com o DPDK, um dos núcleos utilizou sempre 100% de processamento enquanto que os núcleos utilizados pelas máquinas virtuais se mantiveram com utilização semelhante aos dos outros testes. Com o Netmap, verificou-se um uso maior de CPU nas máquinas virtuais, possivelmente causado pela utilização dos drivers especiais do Netmap.

4. Conclusão

Esta pesquisa teve como objetivo a apresentação, análise e comparação de aceleradores de processamento de pacotes. Foi possível identificar claramente vantagens e desvantagens no uso dos *frameworks* para aplicações com alto consumo de recursos de rede. Como vantagem, pode-se considerar o desempenho superior para as métricas avaliadas, enquanto que as desvantagens são relacionadas a dificuldade de configuração e um consumo maior de recursos do sistema como um todo.

Desta forma, acredita-se que, no caso de funções virtualizadas de rede e serviços de alto desempenho, *frameworks* se mostram como uma opção superior quando comparado à pilha do Linux.

Já para o uso com aplicações que não necessitam de alto desempenho de rede, deve-se avaliar as vantagens e desvantagens em usar estes *frameworks*, que podem consumir recursos que não são necessários nestes cenários, diminuindo a quantidade restante para ser usada por outras aplicações. Estas tecnologias podem contribuir para que sistemas tradicionais funcionem em conjunto com *middleboxes*, em ambientes de *datacenter*.

Por fim, como trabalhos futuros, o teste com outros cenários (*e.g.*, máquinas físicas) pode identificar o ganho de desempenho em ambientes diferentes. No entanto

isto requer uma infraestrutura que esteja preparada para receber e implantar aceleradores de processamento de pacotes.

Referências

Bradner, S. and McQuaid, J. (1999). Benchmarking methodology for network interconnect devices. RFC 2544, RFC Editor.

García-Dorado, J. L., Mata, F., Ramos, J., Santiago del Río, P. M., Moreno, V., and Aracil, J. (2013). *High-Performance Network Traffic Processing Systems Using Commodity Hardware*, pages 3–27. Springer Berlin Heidelberg, Berlin, Heidelberg.

Hermsmeyer, C., Song, H., Schlenk, R., Gemelli, R., and Bunse, S. (2009). Towards 100g packet processing: Challenges and technologies. *Bell Labs Technical Journal*, 14(2):57–79.

Intel (2014). Data plane development kit. *URL http://dpdk.org*. Acesso em 27 abr. 2017.

ITU (2016). Ict facts and figures 2016. *URL http://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2016.pdf*. Acesso em 03 set. 2016.

Linux (2005). Packet mmap. *URL kernel.org/doc/Documentation/networking/packet mmap.txt*.

Rizzo, L. (2012). netmap: A novel framework for fast packet i/o. In *2012 USENIX Annual Technical Conference (USENIX ATC 12)*, pages 101–112, Boston, MA. USENIX Association.

Salim, J. H. (2005). When napi comes to town. In *2005 Linux Conf*.

Salopek, D., Vasić, V., Zec, M., Mikuc, M., Vašarević, M., and Končar, V. (2014). A network testbed for commercial telecommunications product testing. In *2014 22nd International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pages 372–377.

Sherry, J., Hasan, S., Scott, C., Krishnamurthy, A., Ratnasamy, S., and Sekar, V. (2012). Making middleboxes someone else's problem: Network processing as a cloud service. In *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, SIGCOMM '12, pages 13–24, New York, NY, USA. ACM.

Tsiamoura, K., Wohlfart, F., and Raumer, D. G. (2014). A survey of trends in fast packet processing. In *Proceedings of the Seminars Future Internet (FI), and Innovative Internet Technologies and Mobile Communication Networks (IITM)*, pages 41–48.

Turner, D. and Chen, X. (2002). Protocol-dependent message-passing performance on linux clusters. In *Cluster Computing, 2002. Proceedings. 2002 IEEE International Conference on*, pages 187–194. IEEE.

Uma arquitetura para Sensoriamento e Tratamento de Eventos voltada à Área de Segurança para Controle e Rastreio de Usuários em Ambientes Físicos

Walther F. Pedrozo[1], Alexandre Silva Rodrigues[2], Bruno S. Alves[2],Clóvisson L. Rosa[1],Tiago A. Rizzetti[1]

[1]Colégio Técnico Industrial de Santa Maria (CTISM) – Universidade Federal de Santa Maria (UFSM), CEP– 97.105-900 – Santa Maria – RS – Brasil

[2]Centro de Tecnologia (CT) – Universidade Federal de Santa Maria (UFSM), CEP-97.105-900 – Santa Maria – RS – Brasil

{waltherpedrozo,clovisson.lopes,alexandre.rodrigues}@redes.ufsm.br, rizzeti@gmail.com,bdalves@inf.ufsm.br

Abstract. *Physical environments of educational institutions and workplaces are increasingly equipped with high quality materials and with high financial value, requiring entry control only from authorized persons. This architecture proposes to solve this problem by implementing a system of control and tracking users in physical environments, which facilitates those responsible for managing the environments to manage the access of persons with authorization.*

Resumo. *Ambientes físicos tanto de instituições de ensino como de locais, trabalhos estão cada vez mais equipados com materiais de alta qualidade e com alto valor financeiro, sendo necessário o controle de entrada somente de pessoas autorizadas. Esta arquitetura se propõem a solucionar este problema implementando um sistema de controle e rastreio de usuários em ambientes físicos, que facilite aos responsáveis pela administração dos ambientes gerenciarem o acesso de pessoas com autorização.*

1. Contextualização

Na maioria das instituições existem ambientes físicos onde se deseja restringir o acesso somente para pessoas autorizadas. O sistema de chaves físicas apresenta falhas de segurança, pois não possui, vinculado a si próprio, dispositivos que limitem o acesso de um usuário não autorizado, nem auditoria de uma forma eficiente.

Para impossibilitar o acesso a de pessoas indevidas a esses ambientes, torna-se necessário um método que permita o acesso somente para pessoas cadastradas, registrando inclusive as tentativas de acesso. É desejado que o método de acesso possua mecanismos de registro e controle, preenchendo as prerrogativas de protocolos da família AAA, autenticação, autorização e auditoria [Santos, 2007].

Outro fator importante é a centralização das informações do sistema. Dessa forma, é possível solucionar o problema das fechaduras eletrônicas stand, mantendo um controle sobre diversos dispositivos e ambientes, utilizando uma base central para autenticação, autorização e auditoria dos dispositivos do sistema. Com base nisso, este trabalho apresenta aprimoramentos no controle de acesso implementado na plataforma

ESC (*Environment Security Control*) que permite controle de acesso a ambientes, gerenciando permissões e auditoria, baseado na interação entre vários dispositivos físicos e um software gerente. Esse, por sua vez, caracteriza-se como um middleware para tratamento dos eventos gerados pelos diversos dispositivos integrados nesta plataforma. A principal mudança em relação a anterior foi a substituição da plataforma Arduino, que apresentou problemas de instabilidade e baixa capacidade processamento, pelo Raspberry Pi, o qual soluciona esses problemas apresentados em função da utilização de um hardware com maior capacidade computacional.

2.Trabalhos Relacionados

O controle de acesso apresenta uma série de problemas a serem solucionados. Nesse contexto, é possível encontrar diversas propostas que visam solucionar problemas semelhantes, utilizando diferentes abordagens. Entre as soluções acadêmicas, podemos destacar:

Sentinel: um engenho Java para controle de acesso RBAC, este trabalho descreve um controle de acesso baseado em papéis (RBAC), que pode atuar de forma genérica em diferentes tipos de aplicações. A autorização baseia-se na atribuição de diferentes tipos de privilégios para cada grupo de usuários. Embora exista um módulo de auditória, esse não é muito desenvolvido, visto que, a principal preocupação dessa proposta é realizar o processo de autenticação. [Mattos, 2003].

Sistema de Controle de Acesso Utilizando Dispositivos Embarcados, apresenta uma proposta para o controle de acesso às salas de aula de uma universidade. Esse sistema é constituído por um middleware de um servidor de autenticação. O middleware é responsável por realizar a autenticação de usuários por meio da leitura de tags RFID e enviar para o servidor de autenticação, utilizando a rede *ethernet*. O servidor de autenticação atua de forma centralizada, atendendo as requisições de todos os middlewares. O autor ressalta a necessidade de uma interface amigável para facilitar a administração do sistema, embora não esteja implementada [Peixoto, 2013].

A Arquitetura ESC diferencia-se das citadas anteriormente, por realizar o registro de todos os eventos tratados, possui uma interface de interação com o sistema e atua de forma centralizada sobre dispositivos físicos que integram o sistema.

Além disso, existem soluções proprietárias para resolver estes problemas. Entre elas, podemos citar:

Controle de Acesso Remoto Siemens: permite cadastrar usuários, agendar horários para cada usuário e possui características de auditoria, informando algum acesso em horário indevido. Além disso, é possível vincular esse sistema a um sistema de alarme monitorado 24 horas [Heinsch, 2011].

Controle de acesso NibTec: possibilita que diferentes dispositivos de entrada sejam utilizados, como por exemplo: teclados numéricos, cartões com tecnologia de radiofrequência (RFID) e dispositivos biométricos. Além disso, permite realizar auditorias, gerando relatórios sobre entrada e saídas de usuários [Heinsch, 2011].

Por tratar-se de soluções proprietárias, as empresas não disponibilizam maiores informações sobre os códigos-fonte e apresentam um custo maior para serem adquiridos pelo usuário final. Nesse aspecto a Arquitetura ESC se destaca, utilizando plataformas

abertas para desenvolver os *hardwares* e *softwares*, permitindo alterações e adaptações para a utilização em diferentes ambientes.

3. Arquitetura ESC

A arquitetura ESC foi projetada com vistas a proporcionar uma plataforma para monitoramento de ambientes físicos, realizando tratamento de eventos via sensores e atuação no ambiente através de atuadores. É portanto um sistema de tratamento de eventos onde através de informações lógicas controla-se parâmetros do ambiente físico.

A arquitetura ESC é implementada sob 3 diferentes componentes, o ESCMA (ESC *Manager*), ESCHA (ESC *Hardware*) e ESCI. (ESC *Interface*) Na figura 1 são apresentados esses componentes e a relação existente entre eles.

O ESCHA é um conjunto de sensores e atuadores conectados a um microcontrolador , com capacidade de reconhecer eventos e transmitir e/ou receber informações através de uma rede de comunicação baseada em protocolo IP. O ESCMA é um software de gerenciamento que atua de forma centralizada, sendo responsável pela comunicação com todos os dispositivos que compõem o sistema, realizando a coleta de eventos e seu respectivo tratamento. O ESCI realiza a interação entre o gerente (ESCMA) e o usuário do sistema, podendo configurar módulos, conexões e o gerenciamento das conexões com o banco de dados.

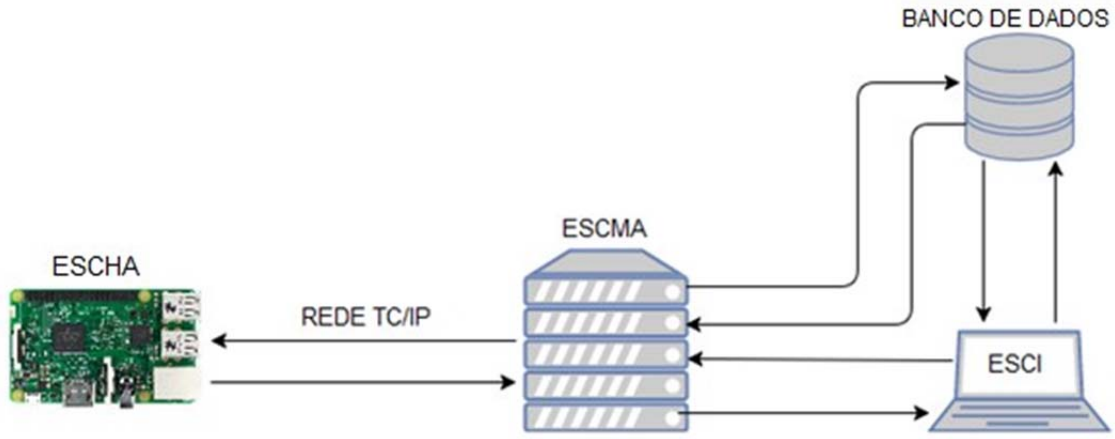


Figura 1. Arquitetura ESC

4. Comunicação

A comunicação entre o ESCHA e o ESCMA utiliza a pilha de protocolos TCP/IP. Esta oferece as funcionalidades necessárias para o envio de dados, eliminando a necessidade de uma nova rede específica.

Além disso, é necessário um protocolo de comunicação para enviar e receber mensagens, implementado em ambos os subsistemas, de maneira que seja possível o entendimento de uma informação enviada. Dessa forma, vários critérios devem ser observados, como por exemplo: a troca de mensagens deve ser sincronizada, necessidade de uma forma de criptografar as mensagens, possibilitar que qualquer dispositivo possa ser utilizado como sensor ou atuador e respeitar as limitações de processamento do hardware desenvolvido.

O protocolo desenvolvido baseia-se no envio de uma string, conforme apresentada na Figura 2. Essa string é montada conforme um padrão adotado nos subsistemas. Para realizar a comunicação é utilizado o *modelo cliente-servidor*, nesse cenário o ESCHA atua como o cliente, que envia uma solicitação ao servidor (ESCMA), que realiza o processamento dos dados e envia uma resposta. Visando manter atualizado na interface o estado do gerente e dos dispositivos, a arquitetura utiliza o mecanismo de *pooling,* no qual é feito o envio de uma mensagem para o ESCMA informando o estado do dispositivo, caso não haja geração de eventos em um intervalo de tempo preestabelecido.

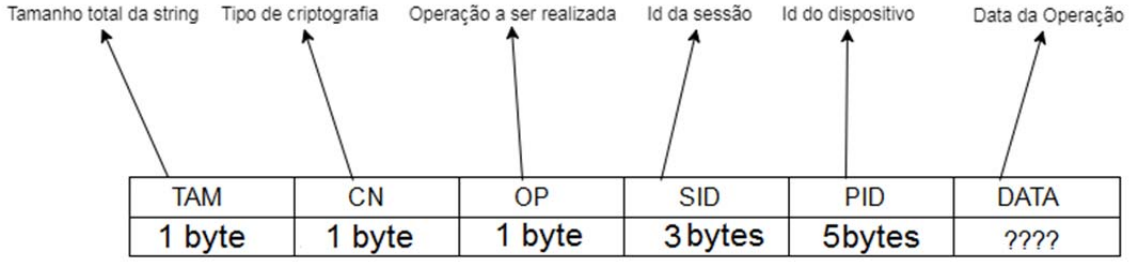


Figura 2. Pacote padrão

4.1. ESCHA

Os primeiros componentes utilizados como dispositivo de hardware na arquitetura ESC foi a plataforma arduino. Porém, através do intenso uso experimentado no estudo de caso de sua implantação, verificou-se que a baixa capacidade computacional gera eventualmente instabilidades na comunicação causando problemas de difícil rastreamento e replicação. Desta forma, optou-se pela experimentação de plataforma de hardware mais robustas, capazes de realizar operações de criptografia e comunicação de rede sem sobrecarga ao sistema, a solução natural foi o emprego da plataforma Raspberry PI 3. A principal contribuição dessa plataforma é facilita a integração entre os dispositivos eletrônicos (sensores e atuadores) a comunicação de rede através da implementação tradicional da pilha de protocolos *TCP/IP* utilizadas por sistemas operacionais de propósito geral, como o Linux.

Uma grande vantagem no uso da arquitetura baseada em Raspberry é que se pode utilizar um sistema de propósito geral, baseado em Linux, capaz de oferecer interfaces de programação e utilização idênticas àquelas encontradas em computadores tradicionais. Isso facilita, além do desenvolvimento em plataformas *cross compile* de fácil simulação, a possibilidade rastreabilidade de bugs de forma mais fácil e eficiente daquela presente na plataforma Arduino (somente informações via serial e, que causam impacto significativo de recursos. Além disso, ela apresenta diversas vantagens em relação a outras plataformas, como processamento, software *open source* e a possibilidade de expansão utilizando as portas GPIO. Na versão dois da arquitetura foi desenvolvida uma placa, que conta com: módulo e antena para leitura do cartão RFID, componentes eletrônicos necessários ao acionamento de uma fechadura eletromagnética, além de Leds indicadores do funcionamento da mesma.

O ESCHA, basicamente pode ser visto como um conjunto de sensores e atuadores, que realiza uma comunicação com um gerente centralizado (ESCMA), por meio de redes TCP/IP.

A arquitetura ESC, por ser escalável torna-se possível a adição de dispositivos físicos sem necessitar alteração no gerente (ESCMA), tornando o sistema mais flexível a mudanças.

4.2 ESCMA

Esse subsistema trata do módulo de gerenciamento centralizado, projetado para armazenar as configurações de todo sistema, provendo escalabilidade, flexibilidade, e facilidade de uso. Portanto estas características permitem constantes modificações e adição de novas categorias de dispositivos físicos, através da adição de plugins.

Nesse contexto, o ESCMA pode ser dividido em módulos: dispositivo físico, gerente, base de dados, auditoria e interface. Desta forma, cada módulo exerce uma atividade especifica, conforme pode ser observado na figura 3.

A conexão com o banco de dados acontece, quando o ESCMA recebe um evento e precisa consultar se o usuário tem permissão de acesso ao ambiente desejado, retornando ao dispositivo físico a resposta. Ela também é utilizada para armazenar registros de acessos permitidos e não permitidos, para utilização em caso de auditoria.

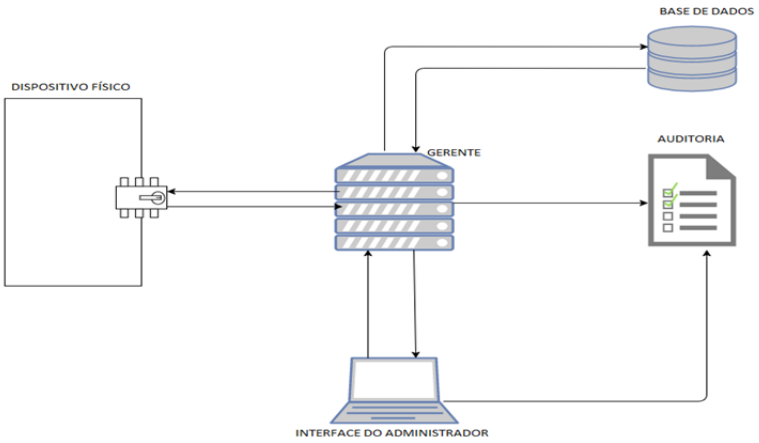


Figura 3. Conexão entre módulos

Na versão um da Arquitetura ESC, a coleta de informação utilizava o método *polling* o qual buscava informações em um intervalo de tempo definido, essa técnica era utilizada devido às limitações do hardware dessa versão, pois impediam que esse funcionasse como um cliente, apresentando limitações no buffer. Visando evitar o desperdício de recursos computacionais a versão dois da arquitetura ESC, modificou o método de busca de informação, agora os dispositivos físicos iniciam comunicação com o gerente somente quando houver alguma alteração no ambiente, sendo assim só haverá comunicação quando um evento precise ser tratado pelo gerente, esse modelo é conhecido como *modelo cliente-servidor*, no qual os dispositivos (ESCHA) são os clientes e o gerente (ESCMA) atua como servidor que espera a comunicação dos clientes.

4.3 ESCI

Com a necessidade de haver comunicação entre o administrador e gerente, foi projetada uma interface web, utilizando a linguagem de programação *PHP* e a linguagem de marcação *HTML* e *CSS*. Essa interface pode coordenar a configuração de todo sistema, podendo interagir. A principal vantagem da interface web é a flexibilidade que a mesma oferece, pois é acessível de qualquer lugar, desde que o usuário possua cadastro e permissões de acesso à página. O ESCI oferece diversas funções aos usuários como adicionar e remover dispositivos de entrada e usuários, verificar o status do gerente, ver os dispositivos existentes, realizar auditorias, gerar relatórios de acesso e tentativas de acesso tanto por dispositivos quanto por usuários.

5. Resultados e Conclusões

Visando analisar o desempenho e confiabilidade da arquitetura proposta, foram feito dois testes: Um utilizando o Arduino e outro à plataforma Raspberry.

Os testes foram realizados em duas sessões de 3 minutos cada. Durante cada uma foram enviadas 10 tags com autorização e 10 sem. Nos testes foram enviadas 40 solicitações de acionamento de dispositivo ao gerente.

Nos testes com o Arduino foi utilizado o cenário onde é feita a passagem de dois cartões RFID nos dispositivos, um com permissão de entrada e outro sem.

Para calcular o tempo decorrido entre a solicitação de abertura e a resposta do gerente, foi feita uma subtração do tempo inicial da solicitação com o de resposta do gerente. Após foi feito a média desses valores, retornando como resultado 1,4784 segundos para o cartão autorizado e 1,2955 segundos para o não autorizado.

No raspberry foram enviadas 40 tags diretamente da plataforma, sendo 20 com autorização e 20 sem. Após houve a adição de uma função no código onde retorna o tempo de resposta do gerente por tentativa e uma média de todas as tentativas. O tempo médio de resposta para tags com permissão foi de 0,19488 segundo e para tags sem permissão de 0,1846265 segundo, tendo uma pequena variação entre os tempos de resposta.

Analisando o Gráfico 1 e 2, é possível perceber que existe uma grande diferença de tempo de resposta na plataforma Arduino e Raspberry, utilizando as tags válidas a diferença foi de 1,283592 segundos. Já em tags inválidas foi de 1,0958735 segundos.

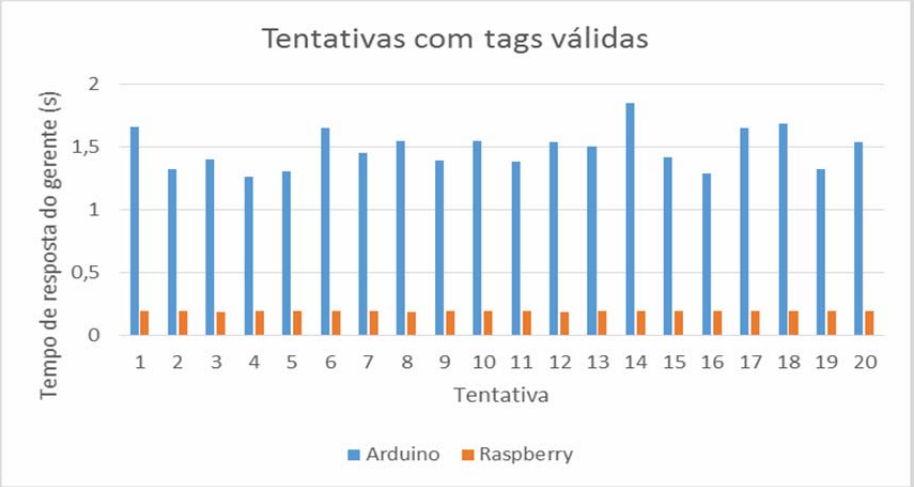


Gráfico 1. Comparação do tempo de resposta do gerente para tags válidas entre as duas plataformas.

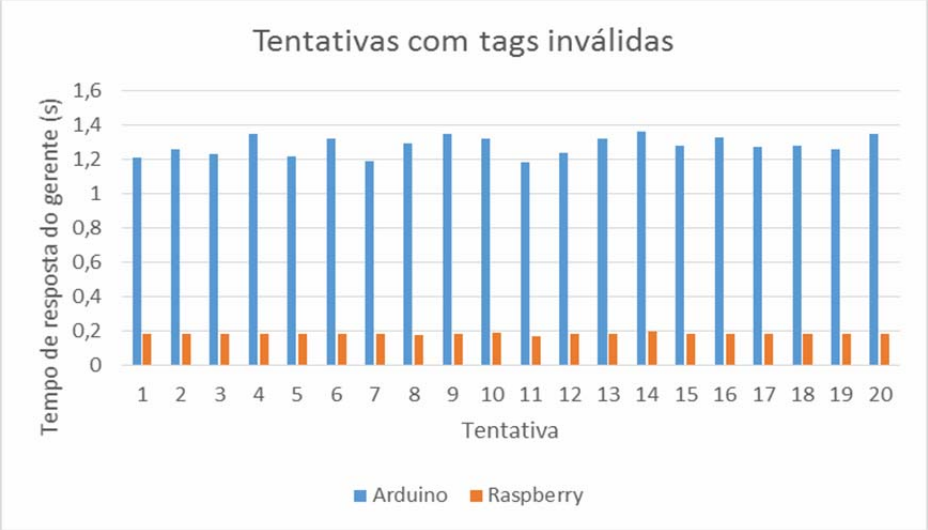


Gráfico 2. Comparação do tempo de resposta do gerente para tags inválidas entre as duas plataformas.

A partir desses testes é possível observar a estabilidade do gerente, pois mesmo tendo sido utilizado diversas vezes em um curto espaço de tempo, não houve problemas em seu sistema, o que comprova o quão escalável e confiável a arquitetura ESC é.

Outro ponto a ser observado é a diferença nos tempos de resposta utilizando as duas plataformas, constata o quão relevante foi à substituição da plataforma Arduino pelo Raspberry, o principal ponto para a diferença no tempo de respostas está no tempo de criptografia, pois o raspberry por apresentar melhores componentes computacionais, sendo assim o tratamento das mensagens é feito de forma mais rápida.

5. Trabalhos Futuros

A partir de agora pretende-se adicionar mais funcionalidades a arquitetura ESC, visto que a mesma já se mostrou bastante flexível. Um dos exemplos de funcionalidades a ser incrementada é o controle da temperatura e umidade do ambiente onde a placa atua, visando evitar danos físicos as placas.

Referências

Gilmore, W. J., (2010), Beginning PHP and MySQL.

Heinsch, L. R. (2011), Sistema automatizado para controle de acesso ás salas do CTISM. Trabalho de Conclusão de Curso, Universidade Federal de Santa Maria.

Hailperin, M. (2007), Operating systems and middleware: interaction.

Mattos, C. L. A. (2003), Sentinel: um engenho Java para controle de acesso RBAC. Trabalho de Conclusão de Curso, Universidade Federal de Pernambuco.

Mendes, Antonio. (2002), Arquitetura de Software: desenvolvimento orientado para arquitetura.

McRoberts, M. (2015), Arduino Básico – 2º Edição.

Monk, Simon. (2016), Movimento, Luz e Som com Arduino e Raspberry; Tradução

Peixoto, T. M. (2013), Sistema de Controle de Acesso Utilizando Dispositivos Embarcados. Trabalho de Conclusão de Curso, Universidade Federal de Juíz de Fora.

Quintas, D. L. (2012), Controle de Acesso Utilizando uma Rede de Microcontroladores. VII Jornada de Iniciação Científica, Desenvolvimento Tecnológico e Inovação do Ifes.

Raguzzoni, J. C. M., Heinsch L. R. e Rizzetti, T. A. (2012), Uma arquitetura para desenvolvimento de dispositivos de autenticação e acesso a espaços físicos.

Robertson Craig, Ronald C. Beavis; TANDEM: matching proteins with tandem mass spectra. Bioinformatics 2004; 20 (9): 1466-1467. doi: 10.1093/bioinformatics/bt092.

Santos, A. (2007), Gerenciamento de Identidades.

Silva, Maurício Samy (2011), CSS3: desenvolva aplicações web profissionais com o uso dos poderosos recursos de estilização da CSS3.

Sugimoto, G.M., Aguiar, L.K., (2011), Implementação de Protocolos da Pilha TCP/IP.

Ferramentas para Mapear e Minimizar o Consumo de Energia em Redes de Sensores Sem Fio

Larissa S. Del Rio[1], Tiago A. Rizzetti[1], Alexandre Rodrigues[2], Luciane N. Canha[2], Marcio de Abreu Antunes[3]

[1]Colégio Técnico Industrial de Santa Maria – CTISM – UFSM
Av. Roraima, 1000 – 97015-900 – Santa Maria – RS – Brasil

[2]Centro de Tecnologia – UFSM – Santa Maria – RS – Brasil

[3]Companhia Estadual de Energia Elétrica – Distribuição – CEEE-D – RS – Brasil

{larissa, alexandre.rodrigues}@redes.ufsm.br, rizzetti@ctism.ufsm.br, lucianecanha@ufsm.br, marcioaa@ceee.com,br

Abstract. *A Wireless Sensor Network (WSN) can be defined as a set of devices capable of collecting information without the presence of a physical structure interconnecting these devices. One of the most critical points in a WSN is power consumption, because the main power source of the network components are batteries. The useful life of the network is conditioned to the duration of the batteries. This paper presents some of the existing solutions in the literature that can minimize and/or optimize energy consumption in a WSN. It is also presented a proposal of hardware to be used in a WSN with the same function, that is, to reduce the energy consumption in the network.*

Resumo. *Uma Rede de Sensores Sem Fio (RSSF) pode ser definida como um conjunto de dispositivos capazes de realizar coleta de informações sem a presença de uma estrutura física interligando-os. Um dos pontos mais críticos em uma RSSF é o consumo de energia, já que a principal fonte de alimentação dos componentes da rede são baterias. O tempo de vida útil da rede fica condicionada ao tempo de duração das baterias. Este artigo tem o objetivo de apresentar algumas das soluções existentes na literatura, que possam minimizar e/ou otimizar o consumo de energia em uma RSSF. Também é apresentada uma proposta de hardware para ser utilizado em uma RSSF com a mesma função, ou seja, diminuir o consumo de energia na rede.*

1. Introdução

Uma Rede de Sensores Sem Fio (RSSF) ou WSN (*Wireless Sensor Networks*) pode ser definida como o uso de inúmeros sensores espalhados geograficamente sem a necessidade do uso de qualquer estrutura física. Entre as aplicações desse tipo de rede destaca-se o monitoramento de ambientes. Um dos usos mais decorrentes estão nas áreas industriais, na coleta de dados em locais perigosos ou de difícil acesso. Logo, o uso de uma WSN pode ocorrer nos mais variados locais, como em uma floresta, por exemplo, onde não há a disponibilidade de energia elétrica [Shelke *et al*. 2013].

Em uma rede de sensores sem fio há o emprego elevado do número de sensores, que são chamados de nós sensores ou nodos. Nessa rede, os sensores são responsáveis pelo monitoramento do ambiente, pela coleta e processamento das informações. Após é feito o envio das informações para o *gateway*, que é responsável por encaminhar as informações coletadas para uma aplicação de monitoramento e controle, como sistemas supervisórios (*Supervisory Control and Data Acquisition* - SCADA), por exemplo. Na Figura 1 encontra-se a arquitetura de rede mais comum utilizada em uma WSN. De acordo com [Shelke *et al*. 2013], a principal fonte de energia de uma WSN é uma bateria, assim o tempo de vida útil dos sensores está condicionada ao tempo de duração das baterias. Como, na maioria dos casos, os nós sensores são instalados em locais inóspitos, recarregar ou substituir as baterias torna-se financeiramente e logisticamente inviável.

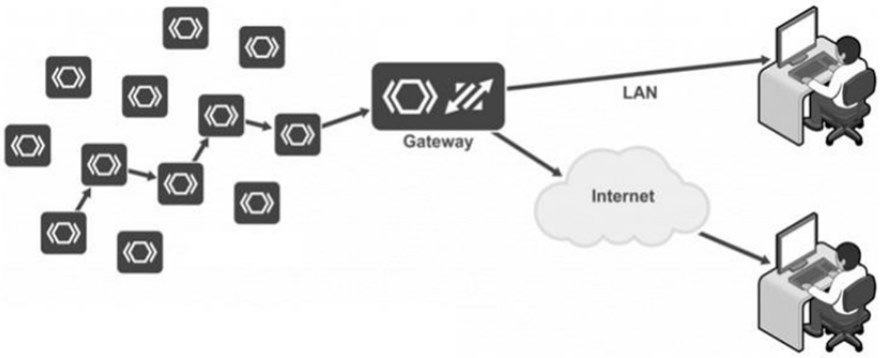


Figura 1. Arquitetura de uma WSN local (adaptado de [micrium.com 2017]).

Devido a essas limitações, na literatura existem algumas técnicas que buscam diminuir o consumo de energia em uma WSN. Esse trabalho apresenta, de forma sucinta, algumas das técnicas utilizadas para minimização do consumo de energia em uma WSN e, também, a proposta de um *hardware*, que minimize o consumo energético, para ser utilizado em uma WSN. No Capítulo 2 desse artigo serão apresentados métodos para mapeamento e minimização do consumo de energia em uma WSN, em seguida, no Capítulo 3 é apresentado o padrão de comunicação 6LoWPAN. No Capítulo 4 é mostrado algumas opções de *hardware* existentes para uso em uma WSN. O Capítulo 5 apresenta uma proposta de um *hardware* equivalente aos apresentados no Capítulo 4. E, por fim, o Capítulo 6 que contém as considerações finais.

2. Mapeamento e Minimização do Consumo de Energia Através de *Software*

Para estudar o consumo energético em uma WSN é fundamental obter o quanto de energia é necessário para manter o funcionamento adequado da rede como um todo ou somente de um nó sensor. Com a obtenção de informações de apenas um nó é possível mensurar o consumo aproximado de toda rede [Souto *et al*. 2005]. Uma das formas mais simplórias de obter informações sobre o consumo de energia em uma WSN é cada sensor enviar mensagens periódicas ao *gateway*, onde a informação contida nas mensagens mostra o quanto de energia disponível cada sensor ainda possui. Porém, a troca de informações é um dos pontos mais críticos quando se trata da questão energética em uma rede de sensores sem fio. Em [Souto *et al*. 2005] é proposta uma solução para esse tipo de abordagem, onde o mapa de energia completo da rede é obtido através de uma amostragem estratificada. Nesse caso, os mapas de energia são

construídos com informações de apenas um subconjunto de nós que compõem a rede. Sendo assim, o consumo de energia necessário para a tarefa é minimizado.

Outra abordagem para o mapeamento de energia é proposta em [Kellner *et al*. 2008], no qual os autores propõem um modelo baseado no mapeamento dos estados de energia do *hardware* dos nós sensores e do sistema operacional. O modelo permite que o sistema operacional do nó sensor seja capaz de mapear o estado do *hardware* ao executar alguma ação como, por exemplo, fazer o envio de uma determinada mensagem ao nó controlador. Dessa forma, essa informação pode ser utilizada para ajustar determinado parâmetro para que haja a redução da energia gasta para a tarefa mapeada.

Para desenvolver técnicas de redução do consumo de energia em uma WSN, primeiramente, deve-se conhecer a arquitetura de um nó sensor. Em [Guidoni *et al*. 2005] é proposto um protocolo para coleta de dados em uma WSN. O protocolo é denominado como TreeDC baseado no algoritmo TopDisc, algoritmo este utilizado para descobrimento da topologia de uma rede de sensores sem fio. De acordo com os autores o TreeDC seleciona os melhores nós da rede através da quantidade de energia que determinado nó possui, assim o roteamento realizado na rede ocorre através da rota que tem a maior disponibilidade de energia. O nó que possui o mapa de energia é denominado como *sink*, esse nó é capaz de predizer se algum nó está próximo de ficar sem energia.

3. O protocolo 6LoWPAN

O IPv6 é um protocolo que atua na camada de rede. Esse protocolo encontra-se em fase de implantação e tem como objetivo substituir o protocolo IPv4. Entre as alterações encontradas no IPv6 pode-se citar o aumento na quantidade de endereços, o endereço IP no IPv6 possui 128 *bits*, ao contrário do IPv4 que possui apenas 32 *bits*; diminuição do custo de processamento de pacotes ao simplificar o cabeçalho com a remoção de alguns campos obrigatórios no cabeçalho IPv4; entre outras [Schrickte, 2013].

Em uma WSN torna-se inviável a utilização do protocolo IPv6 devido a simplicidade dos dispositivos que compõem a rede. Para isso, o 6LoWPAN (IPv6 *over Low Power Wireless Personal Area Networks*) é uma adaptação do protocolo IPv6. O objetivo do 6LoWPAN é possibilitar o uso do IPv6 em redes sem fio de baixa potência, denominadas como LoWPAN (*Low-power Wireless Personal Area Networks*). De acordo com [Montenegro *et al*. 2007], o 6LoWPAN estabelece uma nova camada entre as camadas de enlace e de rede do modelo OSI, demonstrada na Figura 2. Essa camada é denominada como *Adaptation Layer* (camada de adaptação).

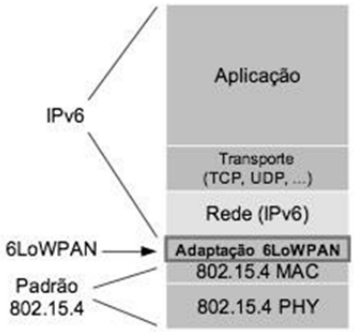


Figura 2. Modelo da pilha de protocolo 6LoWPAN (adaptado de [ipv6.br, 2013]).

A camada de adaptação 6LoWPAN foi desenvolvida para que o protocolo IPv6 pudesse ser utilizado em redes WPAN (*Wireless Personal Area Network*), que contém dispositivos com baixa potência, pouca memória e uma capacidade de processamento limitada. Dessa forma, a camada estabelece algumas alterações no protocolo IPv6, como a redução do cabeçalho dos pacotes IPv6 através de técnicas de compressão, onde os cabeçalhos passam a ter até 2 *bytes*; compressão dos cabeçalhos dos protocolos TCP, UDP e ICMP; fragmentação dos datagramas IPv6; e, a adição de cabeçalhos e informações nos pacotes com o objetivo de otimizar as configurações típicas da rede.

4. O *Hardware* em uma WSN

A escolha do *hardware* correto para compor a rede também deve ser levada em consideração quando a energia é um ponto crítico. É preciso que o MCU (*Microcontroller Unit*) que compõe os nós sensores ou o *gateway* ofereça um baixo consumo de energia, além de suporte aos protocolos utilizados na rede de sensores. Um MCU pode ser definido como um chip que possui uma Unidade Central de Processamento (UCP), memórias, entradas e saídas para utilização externa, periféricos, etc. A seguir serão apresentadas algumas opções de *hardware* encontradas para uso tanto como *gateway* ou nó sensor.

A empresa *Freescale Semiconductor* oferece uma placa para ser usada como nó sensor, denominada como IoT *Low Power Sensor Node board*. Essa placa é composta por um MCU que provê um baixo consumo de energia, o *Freescale* MKW24D512. Possui suporte aos protocolos IPv6/LoWPAN, Zigbee Pro, entre outros. Oferece ao usuário 7 GPIOs (*General Purpose Input/Output*), o que torna possível controlar outros dispositivos conectados à placa. Devido à baixa potência, a placa da *Freescale* pode ser alimentada com apenas uma bateria de 3.0 V e 1200 mAh [NXP, 2015].

Outro exemplo de placa com essa finalidade é o 6LoWPAN IoT *gateway* da fabricante WEPTECH *elektronik* GmbH. A principal funcionalidade da placa é atuar como um roteador de borda em uma rede 6LoWPAN, conectando uma rede IPv6 à *Internet*. O *hardware* é composto por um ARM Cortex-M3 da *Texas Instruments*, o circuito integrado CC2538 que, também, trabalha com uma potência baixa. Este *chip* opera na faixa de 32 MHz e possui uma interface de rádio que opera a 2.4 GHz. Além da interface de 2.4 GHz, a placa possui um *chip* extra, o CC1200 que permite o uso de faixas de frequência menores, ou seja, opera a 868 MHz ou 915 MHz [WEPTECH, 2017].

Ambas as soluções apresentadas utilizam como controladores principais, MCUs que operam em baixa potência. Dessa forma, oferecem uma característica muito importante quando se trata da implementação de uma WSN, o baixo consumo de energia. No próximo capítulo desse artigo será apresentada uma proposta de uma placa, que também constitui um *hardware* para ser utilizado em uma rede de sensores sem fio. Deste modo, a placa possui componentes e especificações similares às mencionadas anteriormente.

5. Proposta de um *Hardware* para Utilização em uma WSN

Esse capítulo apresenta uma proposta de desenvolvimento de uma placa de circuito impresso (PCB), que na prática pode ser utilizada como *gateway* ou como um nó sensor

em uma WSN. Dessa forma, a utilização das técnicas de *software* apresentadas anteriormente agregadas à placa a ser desenvolvida fará com que o *hardware* proposto seja uma escolha promissora para a implementação de uma WSN. Essa PCB tem como controlador principal o circuito integrado (CI) denominado CC2650 (modelo de 48 pinos) da empresa *Texas Instruments*.

Seu processador principal é um ARM Cortex-M3 que opera a 48 MHz. Um dos seus principais periféricos é um controlador para sensores com uma potência ultrabaixa, tornando-se ideal para o uso em WSN, pois é capaz de coletar dados de forma autônoma enquanto o resto do sistema encontra-se suspenso. Pode ser utilizado para aplicações sem fio devido ao suporte a diversos padrões utilizados nesse tipo de comunicação, como, por exemplo, o 6LoWPAN e *Bluetooth*. Essa placa pode ser utilizada em outras aplicações, além da implementação de uma WSN, como o uso na Automação, tanto residencial como industrial. Através das GPIOs (*General Purpose Input/Output*) disponíveis no CC2650, pode-se controlar uma grande quantidade de dispositivos, como sensores e atuadores, por exemplo. Já para a utilização em uma WSN, a principal característica é o pequeno consumo de energia, o que proporcionará um maior tempo de vida útil da rede.

O desenvolvimento da placa tem como base o material disponível pela *Texas Instruments* através da documentação do CC2650. Dessa forma, as especificações necessárias para o funcionamento do CI, como a tensão de alimentação, serão as determinadas pelo fabricante. Porém, pretende-se fazer algumas modificações, como, por exemplo, adicionar um módulo de alimentação separado da PCB principal, já que a tensão de alimentação do CI é muito baixa, entre 1.8 V e 3.0 V. O que torna inviável a utilização da PCB para outras aplicações, onde a tensão pode encontrar-se na casa dos 12 V. Assim, o *hardware* final será composto por dois módulos, onde o primeiro é uma PCB que tem a função de fornecer mais de um nível de tensão (3.0 V e 12 V, por exemplo) e o segundo, o módulo denominado como principal, onde estará o CI (controlador principal) e os demais componentes necessários para o funcionamento adequado da PCB.

O módulo de alimentação será constituído por um circuito eletrônico capaz de reduzir a tensão principal, ou seja, a tensão vinda da fonte de alimentação quando conectado à rede elétrica. Nesse caso, a aplicação que estará inserido o *hardware* será diferente de uma WSN, já que haverá a disponibilidade de alimentação através da rede elétrica do local. Porém, a alimentação também pode ser feita por uma bateria (uso em uma WSN). Para isso, o circuito não irá realizar alteração alguma na tensão de entrada. A tensão que a ser utilizada para alimentar o módulo principal é a tensão da própria bateria conectada ao módulo de alimentação. No módulo principal estará o CC2650, um conector JTAG (*Joint Test Action Group*) de 10 pinos, dois barramentos de pinos (GPIOs disponíveis) totalizando 29, um botão para reset, um borne para alimentação e os demais componentes necessários ao circuito de alimentação do CI principal. Na Figura 3 é apresentado um protótipo inicial do que poderá vir a ser o módulo principal do *hardware* em questão.

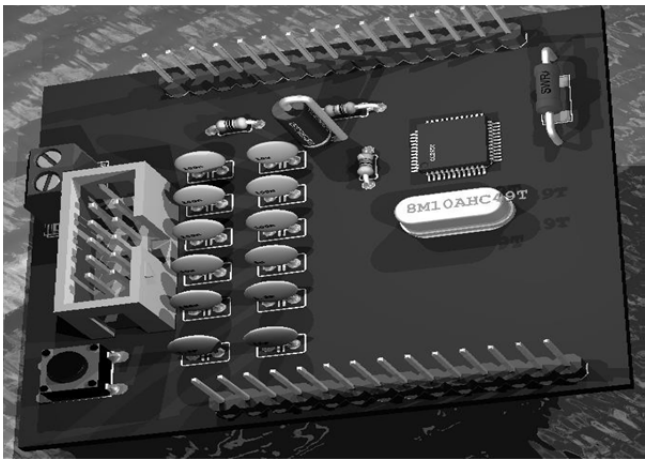


Figura 3. Protótipo do módulo principal [Arquivo Pessoal 2017].

A interface JTAG é responsável pela depuração do CC2650, ou seja, através dela é possível a realização de testes no *hardware*, além de ser possível a programação do microcontrolador. O JTAG implementa uma porta de comunicação serial que dá acesso ao sistema interno de um circuito integrado, sem a necessidade de um circuito externo que permita essa comunicação. Através da interface é possível a transferência do *firmware* (sistema operacional embarcado), *softwares* e dados para a memória não-volátil do dispositivo. Para a programação do dispositivo é necessário a utilização de uma IDE (*Integrated Development Environment*), um ambiente de desenvolvimento integrado. O *Code Composer Studio* é uma IDE desenvolvida pela *Texas Instruments*, que possui suporte aos microcontroladores da fabricante. A IDE possui um conjunto de ferramentas para desenvolvimento e depuração de aplicações embarcadas. Inclui um compilador para as linguagens de programação C/C++, um editor de código fonte, um ambiente com interface intuitiva para a construção de projetos, depurador, entre outros diversos recursos.

O sistema operacional utilizado no CC2650 é um RTOS (*Real Time Operating System*), ou seja, um sistema operacional de tempo real, o qual é destinado à execução de diversas tarefas, onde seus tempos de resposta são pré-definidos. Um RTOS é considerado um sistema mais eficaz, pois, possui uma resposta rápida e previsível a um determinado evento. A *Texas Instruments* possui uma versão própria de um RTOS a ser utilizada em seus microcontroladores. O sistema é denominado como TI-RTOS. Segundo a fabricante, o sistema fornece os componentes de *software* essenciais ao funcionamento do sistema. É uma solução completa em relação aos RTOS, incluindo pilhas de protocolos, comunicação de múltiplos núcleos, drivers de dispositivos e gerenciamento de energia. Há a possibilidade de fazer uso da pilha de protocolos desejada, já que o TI-RTOS suporta diversos padrões que operam na faixa de 2.4 GHz, como *Bluetooth*, *WiFi*, 6LoWPAN, entre outros [*Texas Instruments*, 2017].

De forma genérica, a placa apresentada relaciona-se com o sistema como um todo. A placa envia as informações através da rede 6LoWPAN para o *gateway* 6LoWPAN, que converte o pacote para IPv4 ou IPv6 nativo, após é feita a transmissão desses pacotes através da Internet para, por exemplo, um *software* supervisório.

6. Considerações Finais

O prolongamento do tempo de vida útil de uma WSN depende totalmente da minimização do consumo de energia. Por isso, a literatura está repleta de técnicas que buscam aprimorar a eficiência energética dos nós sensores. No entanto é uma tarefa bastante complexa, pois deve ser levado em consideração diversos fatores e diferentes variáveis para a elaboração dos métodos. Para a implementação dessas técnicas e de diversas outras é necessário um profundo conhecimento da arquitetura de um nó sensor, da árvore topológica da rede, entre outros fatores.

Nesse trabalho foi apresentada a proposta de um *hardware* que tem como objetivo minimizar o consumo de energia em uma WSN. A proposta baseia-se na utilização de um microcontrolador com baixíssimo consumo de energia, desenvolvido especialmente para uso em aplicações onde a energia disponível é um problema crítico. Além disso, o microcontrolador apresenta suporte ao padrão de comunicação 6LoWPAN, que é responsável por tornar possível o uso do IPv6 em uma rede onde os dispositivos disponíveis trabalham com limitada capacidade de processamento e memória. A união das técnicas de minimização apresentas ao *hardware* proposto nesse artigo torna o dispositivo uma eficiente ferramenta para diminuir o consumo de energia em uma WSN. Por exemplo, a integração do *hardware* proposto com o algoritmo TreeDC pode proporcionar uma economia ainda maior de energia, já que o algoritmo escolhe a melhor rota através dos nós que possuem mais carga em sua bateria. Assim, a aplicação desses algoritmos pode trazer um aumento considerável do tempo de vida útil da rede de sensores.

Em comparação com os *hardwares* apresentados, a placa proposta possui algumas vantagens. Entre essas vantagens encontra-se a capacidade de modularização, um módulo responsável pela alimentação, como foi citado anteriormente. Como o desenvolvimento da placa é de responsabilidade dos autores, ela se torna uma opção bastante versátil, já que nas placas mencionadas não há a possibilidade de realizar quaisquer mudanças. Em relação ao consumo de energia dos processadores principais (MCUs), ambas as placas apresentam características semelhantes, já que os MCUs utilizados são de baixa potência, o que minimiza o consumo de energia. Por exemplo, de acordo com as informações contidas nos documentos técnicos, o MCU CC2538 da placa 6LoWPAN IoT *gateway* da fabricante WEPTECH *elektronik* GmbH consome quando o transmissor de informações do rádio está ativo 24 mA de corrente, já o CC2650 utilizado na placa proposto consome realizando a mesma função 6.1 mA. Outro ponto a ser considerado é a quantidade de GPIOs do CC2650 em relação ao MCU *Freescale* MKW24D512, enquanto este possui apenas sete, o CC2650 possui 30 GPIOs.

Após a finalização do desenvolvimento do *hardware* proposto nesse artigo pretende-se realizar testes de desempenho em relação ao consumo de energia, a fim de certificar a eficiência do microcontrolador CC2650 em termos de consumo energético. Também pretende-se desenvolver, futuramente, uma rede de sensores sem fio, onde será implementado o *hardware* a ser desenvolvido e técnicas através de *softwares* para minimização do consumo energético.

Agradecimentos

Os autores agradecem o apoio da ANEEL P&D Código PD-5707-4301/2015, CEEE-D, UFSM e CNPq (Processo 311516/2014-9).

Referências

Dunkels, A., Osterlind, F., Tsiftes, N. and He, Z. (2007). Software-based on-line energy estimation for sensor nodes. In *Proceedings of the IEEE Workshop on Embedded Networked Sensor Systems (IEEE Emnets),* Cork, Ireland.

Guidoni, D., Mini, R. and Machado, M. (2005). TreeDC: Um Algoritmo de Coleta de Dados Ciente da Energia para Redes de Sensores Sem Fio. In *XXV Congresso da Sociedade Brasileira de Computação.* São Leopoldo, RS.

IPv6.br (2013). "*ZigBee usa agora 6loWPAN! Sua próxima lâmpada terá IPv6?*". http://ipv6.br/post/zigbee-usa-agora-6lowpan-sua-proxima-lampada-tera-ipv6/, Agosto de 2017.

Kellner, S., Pink, M., Meier, D. and Blab, E. O. (2008). Towards a realistic energy model for wireless sensor networks. *In Proc. 5th Annual Conference on Wirelles on Demand Network Systems and Services, Garmisch*, 1: 97–100.

Kushalnagar, N.; Montenegro, G.; Schumacher, C. (2007). "*Request for Comments 4919: IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals*", https://www.rfc-editor.org/rfc/pdfrfc/rfc4919.txt.pdf, Julho.

Micrium Partners (2017). "How to think about the Internet of Things (IoT)", https://www.micrium.com/iot/devices/, July.

NXP (2015). "*IoT Low Power Sensor Node Reference Design*", http://www.nxp.com/docs/en/user-guide/ILOSENORDUG.pdf, Agosto.

Sheke, R., Kulkarni, G., Sutar, R., Bhore, P., Nilesh, D. and Belsare, S. (2013). Energy management in wireless sensor network. *UKSim 15th International Conference on Computer Modelling and Simulation,* pages 668-671.

Schrickte, L. F. (2013). "*Projeto, Implementação e Avaliação de Desempenho de Nós e Gateway 6LoWPAN*", https://repositorio.ufsc.br/bitstream/handle/123456789/122627/325160.pdf, Julho.

Texas Instruments (2017). "*CC2538*", http://www.ti.com/product/CC2538/datasheet, Agosto.

Texas Instruments (2017). "*Code Composer Studio™ - Integrated Development Environment*", http://www.ti.com/tool/ccstudio, Julho.

Texas Instruments (2017). "*TI-RTOS: Real-Time Operating System (RTOS)*", http://www.ti.com/tool/ti-rtos, Julho.

Texas Instruments (2017). "*XDS100v2 JTAG Debug Probe (ARM version)*", http://www.ti.com/tool/tmdsemu100v2u-arm, Julho.

WEPTECH (2017). "*SAKER 6LOWPAN IOT GATEWAY*", https://www.weptech.de/en/6lowpan/gateway-saker.html, Agosto.

FERRAMENTA DE ATRIBUIÇÃO INTELIGENTE DE CANAIS EM REDES SEM FIO

Ivania A. Fischer[1], Gabriel Peres Lutz[1], Alisson Perez[1], Bolívar Menezes Silva[1]

[1]Colégio Técnico Industrial de Santa Maria – Universidade Federal de Santa Maria (UFSM)
Caixa Postal 97.001 – 97.105-900 – Santa Maria – RS – Brasil

{ivaniafischer,gabiplutz,alissonperez,bolivar}@redes.ufsm.br

Abstract. *The fall in the price of WiFi equipment and the ease of implementation of these systems, catch the attention of users. Consequently the emergence of problems related to wireless communication, become increasingly common. Such as the excess of access points in small physical areas, which can generate interference between transmission channels of such equipment. Thus, this work presents the development of a tool that aims to minimize the interference caused by overlapping transmission channels between the access points. For the validation of the tool, tests were performed in some scenarios. The results obtained were satisfactory, reaching the proposed objective.*

Resumo. *A queda do preço de equipamentos WiFi e a facilidade de implementação desses sistemas, chamam a atenção dos usuários. Consequentemente o surgimento de problemas relacionados a comunicação sem fio, tornam-se cada vez mais comuns. Como o excesso de pontos de acesso em pequenas áreas físicas, que podem gerar interferência entre canais de transmissão desses equipamentos. Dessa forma, esse trabalho apresenta o desenvolvimento de uma ferramenta que tem por objetivo minimizar a interferência causada pela sobreposição de canais de transmissão entre os pontos de acesso. Para validação da ferramenta, foram realizados testes em alguns cenários. Os resultados obtidos se mostraram satisfatórios, alcançando o objetivo proposto.*

1. Introdução

Segundo [Branquinho 2014], a partir de descobertas científicas, o uso de ondas eletromagnéticas passou ser utilizado como meio de transmissão de dados. Com isso, um aparelho pode conectar-se a uma rede de dados através do ar, sem necessidade de fios. A possibilidade de transmitir informações por ondas desencadeou novas tecnologias, dentre elas a *WiFi*. Os benefícios adquiridos com essa descoberta científica, como facilidade e velocidade de transmissão, foram afetados quando o uso da tecnologia se popularizou.

Segundo a [SYMANTEC 2003], o uso das redes sem fio multiplicam-se cada vez mais ao passo que os preços dos equipamentos tornam-se mais acessíveis ao público. A facilidade de aquisição e implementação de sistemas *Wireless* pode tornar-se um problema. Essas situações desencadeiam problemas nesse tipo de tecnologia. Causados na maioria das vezes por aglomerações dos pontos de acesso em pequenas áreas, por exemplo, o campus de uma universidade, em um condomínio de moradia ou em uma empresa. Em ambientes como esses muitas vezes é necessário o uso de redes sem fio. Aglomerações dos pontos de acesso desencadeiam problemas, como interferências entre

os pontos de acesso e tráfego lento pelo fato de que o canal de transmissão está sobrecarregado nessas áreas.

As adversidades expostas pretendem ser minimizadas com a ferramenta apresentada nesse trabalho, a qual tem o objetivo diminuir problemas de transmissões lentas desencadeadas pelo uso excessivo de canais de transmissão que se sobrepõem. Para que isso seja possível, serão realizadas análises nos canais de transmissão, a fim de detectar presença de interferências e tentar diminuir as mesmas, dentro de um mesmo domínio administrativo, ou seja são redes as quais não possuem um administrador. Para isso, esse trabalho apresenta o desenvolvimento de uma ferramenta que desempenhe esse papel, onde a mesma é subdividida em módulos que irão coletar dados dos pontos de acesso, analisar esses dados e propor melhoria caso seja necessário.

O presente trabalho esta estruturado na seguinte forma: a seção 2 apresentará um breve estudo sobre definições importantes para o entendimento desse trabalho. A seção 3 irá desenvolver um estudo sobre trabalhos relacionados na área. A seção 4 apresenta como a ferramenta foi desenvolvida. A seção 5 apresenta os testes e resultados obtidos. A seção 6 apresenta a conclusão do trabalho.

2. Referencial Teórico

Essa seção tem o objetivo de demostrar uma breve explicação referente a termos utilizados no decorrer desse trabalho. Na subseção 2.1, desenvolve uma breve explicação sobre redes locais sem fio, a subseção 2.2, designada para explicações sobre pontos de acesso e a subseção 2.3, desenvolve um estudo sobre *site survey*.

2.1. WLAN(*Wireless Local Area Network*)

Segundo [Tanenbaum 2003], após o surgimento dos computadores, um dos objetivo de grande parte das pessoas era conseguir entrar em salas, escritórios e seu computador conectar à internet automaticamente. Fato esse que logo tornou-se realidade, com o surgimento das LANs sem fio.

O crescente aumento e inovação da WLAN desencadeou várias atualizações do protocolo 802.11, as quais possuem o objetivo de aprimorar o desempenho da tecnologia WLAN. Conforme [ENGST 2005] menciona, por volta de 1999 o IEEE (*Institute of Electrical and Electronics Engineers*) finalizou o padrão 802.11b. Em 2002, foi distribuído ao mercado o 802.11a, sendo incompatível com o padrão 802.11b e no mesmo ano implementou-se o padrão 802.11g. Atualmente, a nova versão é o 802.11ac. Os padrões WLAN, segundo [Luiz 2015], utilizados no Brasil são os: IEEE 802.11a, 802.11b, 802.11g e 802.11n. Onde o 802.11n e 802.11g são os mais utilizados, e operam na banda ISM (*Industrial Sientific and Medical Band*) de 2,4 GHz.

Na WLAN uma questão bastante comum é a interferência geradas por sinais provenientes de ondas eletromagnéticas de pontos de acesso próximos. Dois tipos de interferências podem ser consideradas, interferência adjacente, causada pela utilização de canais de comunicação próximo ao canal transmitido, interferência co-canal, proveniente da reutilização do mesmo canal pelos pontos de acesso.

2.2. Ponto de acesso (AP)

Segundo [Alecrim 2004] um AP(*access point*) é o equipamento responsável por fazer a interconexão entre os dispositivos móveis em uma rede sem fio. Uma prática comum é

a interligação de um *access point* a uma rede cabeada para, por exemplo, prover acesso à internet e a uma rede local de computadores. Basicamente, um ponto de acesso tem a missão de distribuir o sinal proveniente do cabo. Sem ter a preocupação de realizar o roteamento do mesmo. Portanto, analogicamente, um ponto de acesso tem o objetivo de criar um *link* do cabo ao dispositivo.

As configurações de pontos de acesso, geralmente, são realizadas por uma interface de gerenciamento, essa interface apresenta inúmeras configurações presentes em um ponto de acesso. Por exemplo, um AP na maior parte dos casos apresenta um canal setado pré definido, no entanto, o administrador do ponto de acesso pode alterar esse valor pela interface. Muitas vezes as interfaces dos pontos de acesso são suficientemente intuitivas para essas configurações não se tornarem um problema ao administrador.

2.3. *Site Survey*

Os *access points* atualmente, em sua grande maioria, têm a opção de repetidor, também designada como WDS (*Wireless Distribution System*). Portanto, nota-se que se um AP tem essa característica ele pode apresentar acoplado a isso o *site survey*. Por exemplo, segundo a [Intelbras 2015], seus *access points* tem suporte a WDS, então, ao selecionar a opção de repetidor a busca de *Site Survey* pode ser realizada.

Segundo [PINHEIRO 2003], *site survey* pode ser definido como uma espécie de análise do local (coleta de dados da rede existente, dados como numero de pontos de acesso, canais, potência, alcance) o qual se deseja instalar uma nova rede. Ou também pode avaliar a infraestrutura existente da rede a fim de melhorar a mesma e identificar soluções de possíveis problemas da rede. Portanto, o *site survey* pode ser utilizado tanto para implementação de uma rede nova, como também para buscar dados de uma rede existente. Como visto várias marcas trazem suporte a *site survey* e consequentemente uma maior disponibilidade de captação de dados para gerenciar uma rede.

3. Trabalhos relacionados

No trabalho desenvolvido por [Souto and Pazzi R. 2016], a técnica de seleção de canal é a atribuição dinâmica de canal em redes não coordenadas, ou seja, a técnica é utilizada independente em cada rede. Uma rede não está vinculada a outra para a realização da seleção do canal, o que faz com que a técnica trabalhe de forma descentralizada e não necessite de um controlador central. No entanto, o objetivo de [Souto and Pazzi R. 2016] é conseguir aplicar a seleção de canal em lugares, como em redes residenciais. Em seu trabalho não é levado em consideração pontos de acesso pertencentes a mesma rede. Para calcular o melhor canal, é feito uso de um modelo de interferência denominado FSI (Fator de Sobreposição e Intensidade), o qual pontua os canais conforme sua utilização de forma independente em cada AP. A técnica utilizada realiza a análise de canal em redes não coordenadas, em apenas um AP, e não no conjunto de APs vizinhos.

Segundo [BALBI 2012] a SCIFI (Sistema de Controle Inteligente de Redes sem Fio) é uma ferramenta que integra a técnica de seleção de canal baseada em coloração de vértices, onde o problema de seleção de canal se transforma em um problema de pigmentação de vértices conforme a relevância de cada AP. O desenvolvimento de seleção de canal da SCIFI foi baseado em um algoritmo descentralizado e determinístico que se baseia no grau de saturação. Como a intenção da SCIFI era aplicar o algoritmo em uma

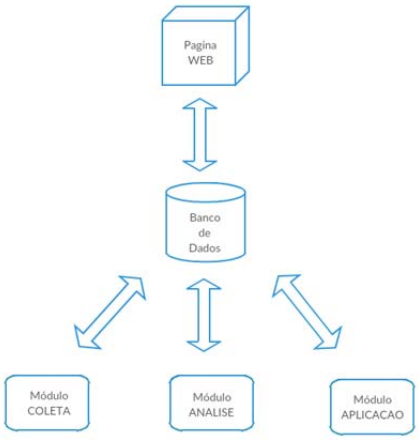
rede centralizada, modificou o algoritmo para sua necessidade. No entanto, ao modificar o algoritmo se deteve os três canais não sobrepostos do 802.11g (1, 6, 11), que podem ser integrados na rede. Essa característica pode ser um ponto negativo pelo fato de que os três canais não sobrepostos dentro da rede podem ser utilizados excessivamente pelas redes vizinhas, o que faz com que as interferências nesses canais sejam muito maiores do que nos canais intermediários entre elas.

Os trabalhos apresentados tratam de seleção de canal em redes não coordenadas, exceto o de [BALBI 2012]. No entanto, esse trabalho objetiva o desenvolvimento de técnicas de seleção de canal em redes centralizadas a fim de colaborar para que o administrador da rede consiga obter o melhor desempenho da mesma. Para isso a análise do melhor canal, diferente do trabalho de [BALBI 2012], irá analisar o melhor canal limitados nos 11 canais disponíveis no Brasil.

4. Metodologia da ferramenta

A ferramenta apresentada analisa os canais de transmissão de pontos de acesso dentro do mesmo domínio administrativo, é estruturada conforme a Figura 1.

Figure 1. Arquitetura da ferramenta



Fonte: Acervo Pessoal

1. A página *web* tem o objetivo de auxiliar o administrador no gerenciamento dos canais dos pontos de acesso que compõem a rede. A mesma possui três funções. A primeira é a de inserir os pontos de acesso da rede no banco de dados a fim realizar as análises. A segunda função é a de executar a análise dos pontos de acesso. E a terceira é alterar os canais conforme a última execução de análise realizada.

2. O banco de dados é o que intermedia as ligações dos módulos, ao mesmo tempo que informa dados relevantes ao administrador pela interface *Web*. O mesmo conta com três tabelas principais que relacionam-se. Uma chamada de AP, outra de DADOS e por fim APLICACAO.

3. O módulo COLETA é o responsável por buscar informações referentes a cada *access point* inseridos pelo administrador e guardar os dados adquiridos no banco de dados. Os dados são coletados através de um algoritmo desenvolvido em Python o qual tem o objetivo de conectar nos pontos de acesso via SSH executar o comando *site_survey* e após salvar os dados adquiridos no banco de dados.

4. O módulo ANALISE tem a função de realizar os cálculos a fim de obter o melhor canal de transmissão. Os cálculos realizados são subdivididos em dois. Esses cálculos são realizados por meio de algoritmos desenvolvidos.

O primeiro cálculo efetuado é o da intensidade resultante, o qual objetiva obter a intensidade que cada ponto de acesso incide em relação a outro ponto de acesso, expresso pela equação 1:

$$IR = RSSI(dBm) - (KdBm) \quad (1)$$

Onde IR é a intensidade que se deseja obter, o RSSI é a intensidade que o AP vizinho incide sobre o AP calculado e o K é definido como -100 dBm, menor nível de sensibilidade de um roteador. Esse valor é designado pela marca dos pontos de acesso, portanto, deve ser consultado na documentação de cada AP.

O segundo cálculo obtém o nível de utilização de cada canal de transmissão do AP calculado. Para isso utiliza-se o cálculo do fator de sobreposição e a Intensidade Resultante. O fator de sobreposição é definido pela seguinte Figura 2, conforme [Souto and Pazzi R. 2016], que utiliza cálculos matemáticos para obter a mesma.

Figure 2. Matriz de sobreposição de canais

Matriz de sobreposição													
Canal	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1	0.77	0.54	0.31	0.09	0	0	0	0	0	0	0	0	0
2	0.77	1	0.77	0.54	0.31	0.09	0	0	0	0	0	0	0	0
3	0.54	0.77	1	0.77	0.54	0.31	0.09	0	0	0	0	0	0	0
4	0.31	0.54	0.77	1	0.77	0.54	0.31	0.09	0	0	0	0	0	0
5	0.09	0.31	0.54	0.77	1	0.77	0.54	0.31	0.09	0	0	0	0	0
6	0	0.09	0.31	0.54	0.77	1	0.77	0.54	0.31	0.09	0	0	0	0
7	0	0	0.09	0.31	0.54	0.77	1	0.77	0.54	0.31	0.09	0	0	0
8	0	0	0	0.09	0.31	0.54	0.77	1	0.77	0.54	0.31	0.09	0	0
9	0	0	0	0	0.09	0.31	0.54	0.77	1	0.77	0.54	0.31	0.09	0
10	0	0	0	0	0	0.09	0.31	0.54	0.77	1	0.75	0.5	0.31	0
11	0	0	0	0	0	0	0.09	0.31	0.54	0.77	1	0.75	0.54	0
12	0	0	0	0	0	0	0	0.09	0.31	0.54	0.77	1	0.77	0.22
13	0	0	0	0	0	0	0	0	0.09	0.31	0.54	0.77	1	0.45
14	0	0	0	0	0	0	0	0	0	0	0	0.22	0.45	1

Fonte: [Souto and Pazzi R. 2016]

A cálculo do fator de utilização de cada canal de transmissão é realizada por meio da seguinte equação 2, a qual utiliza a Intensidade Resultante (IR) e os valores da matriz de sobreposição (Mfs).

$$FT(canal) = \sum_{i=0}^{11} \left(Mfs(canal, i) * \sum_{red}^{n} IR(rede, i) \right) \quad (2)$$

Onde FT é o fator de utilização de determinado canal. Mfs é o valor da matriz de sobreposição de determinado canal e IR é a intensidade resultante. A equação é desenvolvida para cada canal de transmissão que deseja ser verificado o uso. Para isso é realizado o somatório de todos os dispositivos em cada canal e calculado o fator para os mesmos. A partir do fator de utilização consegue-se obter qual o canal com menor fator e assim aplicar as devidas mudanças para que um ponto de acesso opere o melhor possível. Como os cálculos serão realizados em uma rede centralizada, deve ser levado em consideração o que a mudanças de canal de ponto de acesso de um mesmo domínio administrativo afeta nos pontos de acesso ao seu alcance, para isso três possíveis casos de mudança de canais.

- Caso 1:
  A primeira seria um AP independente no domínio administrativo: nesse caso os cálculos de FT são realizados nos 11 canais do AP e identificado o menos sobrecarregado.

- Caso 2:
  A segunda opção seria o caso de haver 2 AP vizinhos no mesmo domínio administrativo. Serão realizados cálculos a fim de decidir qual o melhor canal para cada AP, levando em consideração o quanto a mudança do canal é conveniente para o outro.
  Para resolver esse problema a formulação dos cálculos das mudanças de canal foi baseada em combinações de mudanças entre os APs da rede.
  1ª) alterar o canal do AP1 em relação ao melhor canal do AP2.
  2ª) alterar o canal do AP2 em relação ao melhor canal de AP1.
  A partir do cálculo fator de utilização dos 11 canais cada combinação de cada AP, consegue-se obter o menor fator de utilização de cada combinação. Deve-se nesse momento decidir qual combinação obteve o melhor resultado. Para isso, deve-se descobrir os dois menores valores do conjunto FT, alguns passos são realizados nessa etapa para alcançar o objetivo:
  1º Achar a média aritmética de FT de cada AP.

  2º Excluir os FT acima da média.

  3º Obter o FT com maior número de AP não excluído no cálculo da média.

  4º No caso de empate, escolhe o que tiver o menor FT. Caso os dois forem iguais seria escolhido o primeiro.

- Caso 3:
  Esse caso é composto por conjunto de três pontos de acesso vizinho, o que desencadeia um numero maior de combinações de mudanças de cada AP. Por isso, os cálculos realizados serão limitados em apenas conjuntos de no máximo de três roteadores vizinhos.Não pode-se ter um conjunto de APs com um número de vizinhos na rede superior à três. No caso de haver mais que três pontos de acesso vizinhos é obtido os pontos de acesso mais próximos entre si.
  Após encontrar o conjunto de vizinhos é necessário verificar as combinações de cálculos a serem realizadas. As possíveis combinações nesse caso formam um conjunto de seis possibilidades de análise. Por exemplo, se o conjunto de vizinhos do caso três possuir o AP1, AP2 e o AP3, as possibilidades de combinações serão as seguintes:

  1ª) Alteram o AP2 e AP3 em relação ao melhor canal do AP1;
  2ª) Alteram o AP1 e AP3 em relação ao melhor canal do AP2;
  3ª) Alteram o AP1 e AP2 em relação ao melhor canal do AP3;
  4ª) Altera apenas o canal AP3 em relação ao melhor canal do AP2 e do AP1;
  5ª) Altera apenas o canal AP2 em relação ao melhor canal do AP1 e do AP3;
  6ª) Altera apenas o canal AP1 em relação ao melhor canal do AP2 e do AP3.

5. O módulo APLICACAO será executado caso a escolha do administrador seja a mudança de um canal exposto na página *web* para ele.

Por isso o módulo APLICACAO é executado dentro um código PHP (*Hypertext Preprocessor*) da página. Por exemplo, a página indica ao administrador as mudanças que podem ser feitas para melhorar a transmissão de seus pontos de acesso. Isso, a partir da mudança do canal com menor utilização, sendo esse dado obtido pelos cálculos executados pelo módulo ANALISE.

5. Resultados obtidos

Para realizar a validação da aplicação desenvolvida, foi implementado uma rede de testes, independente da infraestrutura existente no CTISM (Colégio Técnico Industrial de Santa Maria). No cenário desenvolvido é utilizado dois pontos de acesso em duas salas diferentes no prédio de Redes de Computadores do CTISM. Os quais sobrepõem seus sinais um em relação ao outro, portanto são vizinhos de rede.

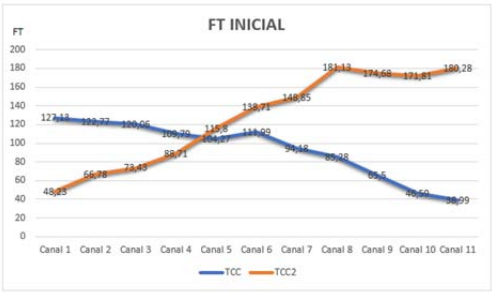
As características que cada ponto de acesso apresenta são as seguintes:

1. AP TCC: IP- 172.17.60.86 Canal- 8 MAC- 90:F6:52:DE:A8:96

2. AP TCC2: IP- 172.17.60.84 Canal- 1 MAC- 90:F6:52:3E:E0:12

Os sinais interferentes de pontos de acesso vizinhos são adquiridos pela busca *site survey*, onde os mesmos são utilizados para análise de canal. O fator de utilização dos 11 canais é mostrado na Figura 3.

Figure 3. Gráfico de FT antes de aplicar a MLWIFI (Melhorias Wifi)

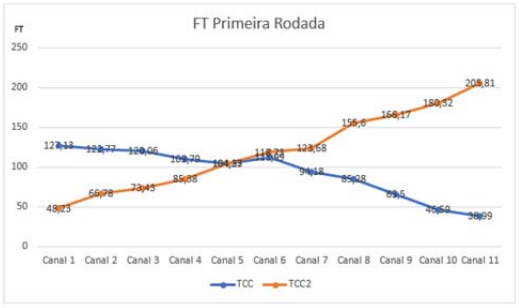


Fonte: Acervo Pessoal

Nota-se que o AP TCC encontra-se no canal 8 que possui um fator de utilização igual a 85,28. Já o AP TCC2 encontra-se no canal 1 que possui um fator de utilização de 48,23.Essas seriam a utilização de cada AP antes da MLWIFI realizar os cálculos de melhoria. Portanto, após a MLWIFI calcular os melhores canais, verifica que a melhor combinação de troca seria modificar o AP TCC2 para o canal 11 e o AP TCC2 manter-se como está. Isso para primeira rodada de análise do módulo Analise. O resultado é mostrado na Figura 4

A partir da análise dos dados dos gráficos, nota-se que a ferramenta conseguiu melhorar o fator de utilização do canal que o AP TCC utiliza cerca de 55,1 % e do AP TCC2 0 % já que o canal não foi alterado e o seu vizinho não foi trocado para um canal sobreposto ao seu. Pode-se verificar que nesse caso a MLWIFI mostrou-se eficiente boa na busca do melhor canal de operação.

Figure 4. Gráfico do FT depois de utilizar a MLWIFI



Fonte: Acervo Pessoal

6. Conclusão

A partir dos testes e resultados apresentados neste trabalho, pode-se observar que houveram reduções significativas na interferência causada por pontos de acesso vizinhos, sejam eles parte do domínio administrativo do administrador da rede, ou dispositivos externos.

A análise de canais foi realizada com base nos dados obtidos em cenários que simulam ambientes reais, na prática, o que torna os testes condizentes com a realidade. Pode-se dizer que os resultados obtidos foram satisfatórios e demonstram que o objetivo da ferramenta foi alcançado.

7. Referências

Alecrim, E. (2004). Diferenças entre hub, switch e roteador. https://www.infowester.com/hubswitchrouter.php.

BALBI (2012). Algoritmo de seleção de canais centralizados para redes iee 802.11 com controlador. H. F. S., Carrano R., D. Saade, C. Alburquerque, L. Magalães.

Branquinho, O. (2014). *Tecnologias de Redes sem Fio.* Rio de Janeiro: Rede Nacional de Ensino e Pesquisa, Escola Superior de Redes.

ENGST, G. F. (2005). *Kit do Iniciante em Redes Sem Fio*. O guia pratico sobre redes Wi-Fi para Windows e Macintosh, São Paulo. Ed.: Pearson Makron Books, 2th edition.

Intelbras (2015). Manual do usuário. http://en.intelbras.com.br/sites/default/files/downloads/manual_do_usu Acessado em junho de 2017.

Luiz, T. A. (2015). Alocação de canal em redes wlan considerando a utilidade marginal total da conexão para usuários. Universidade Federal de Ouro Preto.

PINHEIRO, J. M. S. (2003). Guia completo de cabeamento estruturado.

Souto, M. E. and Pazzi R., J. K. (2016). Atribuição dinâmica de canais em redes sem fio não coordenadas ieee 802.11, baseada em fatores de sobreposição e intensidade de sinal.

SYMANTEC (2003). Implementando uma lan sem fio segura. http://www.symantec.com/region/br/enterprisesecurity/content/framework/BR_3074.html.

Tanenbaum, A. S. (2003). *Computer Networks*. ed. Campus, Holanda.

Redes Ubíquas, uma nova maneira de comunicação transparente e adaptativa: Mapeamento Sistemático

Anderson M. da Rocha[1], Gabriel Marchesan[1], Nilton C. B. da Silva [1] e Thales N. Tavares[1]

[1]Departamento de Computação Aplicada
PGCC – Universidade Federal de Santa Maria (UFSM)
Santa Maria – RS – Brasil

{amonteiro,gmarchesan,nbatista,tntavares}@inf.ufsm.br

Abstract. *Currently the multiple platforms computing, on many different devices and the idea that people always need to transmit and receive any type of information, anytime and anywhere makes studies on ubiquitous networks being expanded. A ubiquitous network consists on a ubiquitous software and hardware arrangement and on different access technologies networks. Several applications are being developed and studied in different areas to meet the information demand anytime and anywhere. In order to discover what has been studied and applied in the ubiquitous networks area, this paper has developed a systematic mapping (SM).*

Resumo. *Atualmente a computação em múltiplas plataformas, em diferentes tipos de dispositivos e a ideia de que as pessoas sempre necessitam transmitir e receber qualquer tipo de informação, a qualquer hora e qualquer lugar faz os estudos em redes ubíquas estar em expansão. Uma rede ubíqua constitui-se de um arranjo de software e hardware onipresentes e de redes com diversos tipos de tecnologias de acesso. Várias aplicações estão sendo desenvolvidas e estudadas em diferentes áreas, para atender à demanda da informação a qualquer hora, em qualquer lugar. Com a finalidade de descobrir o que vem sendo estudado e aplicado na área de redes ubíquas, este trabalho desenvolveu um mapeamento sistemático (MS).*

1. Introdução

Atualmente, com o crescimento da melhoria da telefonia móvel, dos inúmeros aparelhos de conexão de rede sem fio, do aumento do tráfego de dados da banda larga e também da computação em múltiplas plataformas, em diferentes tipos de dispositivos, há a ideia de que as pessoas sempre necessitam transmitir e receber qualquer tipo de informação, a qualquer hora, e em qualquer lugar.

Assim, a ideia de utilizar esse paradigma a nosso favor tem o intuito de trazer ao usuário mais comodidade ao emergir em um ambiente ciente de contexto. Uma rede ubíqua constitui-se de um arranjo de *software* e *hardware* onipresentes e de redes com diversos tipos de tecnologias de acesso. Várias aplicações estão sendo desenvolvidas e estudadas em diferentes áreas, para atender à demanda da informação a qualquer hora, em qualquer lugar.

Hoje em dia, a maioria das pessoas usam a Internet para algum tipo de atividade diária. Para suportar uma crescente demanda do usuário, a Internet evoluiu para incluir não apenas o computador *desktop*, mas também sistemas mais invasivos, como *smartphones* e dispositivos embarcados que são encontrados em edifícios inteligentes, veículos e em vários tipos de equipamentos. A popularidade dos sistemas de comunicação em rede agrava os problemas relacionadas com a rede e gerenciamento de dados, devido à sua mobilidade, disponibilidade intermitente, bem como *software* e *hardware*. Estas questões são mais relevantes quando os usuários passam a ter um papel ativo como uma entidade de rede, e não apenas como consumidor e produtora de dados Sofia *et al* (2012).

Com esses conceitos os estudos sobre redes ubíquas está cada vez mais em expansão e há vários paradigmas a serem estudados, melhorados, qualificados e/ou (re)criados, levando em conta a exigência de mobilidade e adaptação da interconexão das pessoas com seus dispositivos e de outrem.

Conforme afirma Atzori *et al* (2010) hoje as pessoas vivem em ambientes totalmente conectado, em sua maioria usando dispositivos móveis, bem como dispositivos fixos, sem restrição de tempo, lugar e sistema. As informações são disponíveis on-line, e, portanto, acessível através das redes também, a qualquer tempo, a qualquer lugar e com quaisquer dispositivos.

Nessa convergência, o trabalho de Weiser (1991) aborda que a Computação Ubíqua permite que as pessoas e o ambiente, com a combinação de várias tecnologias computacionais, realizem a troca de informações e serviços a qualquer hora e em qualquer lugar. A utilização desta tecnologia em diversas áreas objetiva facilitar a interação do usuário com as aplicações computacionais.

Com a finalidade de descobrir o que vem sendo estudado e aplicado na área de Redes Ubíquas, foi desenvolvido um mapeamento sistemático, onde será apresentado os resultados obtidos e uma análise dos estudos encontrados na literatura.

O artigo está organizado da seguinte forma. Na Seção 2 aborda os conceitos sobre o tema Redes Ubíquas/Pervasivas. A Seção 3 traz os trabalhos correlatos. A Seção 4 contém a metodologia utilizada para realizar a pesquisa. Na Seção 5 é apresentado o mapeamento sistemático, descrevendo o processo de busca, bem como os critérios utilizados para montar a estratégia de busca. Na Seção 6 são apresentados os resultados, e por fim na Seção 7 as considerações finais.

2. Redes Ubíquas/Pervasivas

Redes pervasivas consistem em uma combinação de hardware e software ubiquamente incorporados em dispositivos em nosso ambiente, em múltiplas redes de diferentes operadores de rede com diferentes tecnologias de acesso. Esses sistemas requerem colaboração, e envolve interações complexas entre pessoas, objetos inteligentes e tecnologia. Representam a disponibilidade de recursos de computação e comunicação invasivos McCann e Sterritt (2010).

Rede ubíqua representa a disponibilidade de recursos de computação e comunicação difundidas. Redes ubíquas consistem em várias redes de diferentes operadores de rede com diferentes tecnologias de acesso. Isto leva a aumentar as

tendências de comunicações de rede onipresente como os usuários têm a liberdade de escolher as tecnologias de acesso, aplicações e serviços Yeun *et al.* (2005).

Segundo Thabo *et al*. (2011), redes ubíquas/pervasivas podem geralmente ser classificados em duas categorias, que pode estar dentro da rede (LAN) ou através de redes. A LAN ubíqua/pervasiva é quando se tem uma LAN de dispositivos diferentes, por exemplo: Windows, Linux, Mac PCs. Uma rede ubíqua/pervasiva pode também ser formada através de redes (LANs), utilizando diferentes tecnologias, como por exemplo; *wireless*/*Ethernet* LAN conectada a uma rede celular. Estas redes se conectam para formar uma WAN ubíqua/pervasiva, Obaidat *et al.* (2011).

3. Trabalhos Correlatos

Existem vários trabalhos que utilizam o Mapeamento Sistemático (MS) para melhor estabelecer uma visão aprofundada sobre determinado assunto. Neste trabalho balizou-se nossa ideia com as pesquisas realizadas por Abreu *et al.* (2012), Borges *et al.* (2013) e Magalhães *et al.* (2013).

O trabalho desenvolvido por Abreu *et al.* (2012) tem como objetivo identificar tecnologias que auxiliam o desenvolvimento de softwares para a educação. Os autores buscaram conceitos sobre software educacional, fazendo o levantamento do referencial teórico e da revisão bibliográfica do assunto em questão, montando um protocolo de pesquisa como guia para concluir o MS.

Já Borges *et al.* (2013), tinham como objetivo saber sobre em qual contexto e áreas da educação a gamificação foi mais explorada, os tipos de estudos realizados e quais técnicas de gamificação foram estudadas. Para atender a estes objetivos, os autores se basearam no processo descrito por Petersen *et al.* (2008), onde contém cinco passos importantes a serem seguidos para realizar o MS: (i) definição de questões de pesquisa, (ii) realização da pesquisa de estudos primários relevantes, (iii) triagem dos documentos, (iv) *keywording* dos resumos, e (v) a extração de dados e mapeamento.

Outro trabalho é de Magalhães *et al* (2013), identificaram como se caracteriza a pesquisa em informática na educação no Brasil com base nas publicações do SBIE (Simpósio Brasileiro de Informática na Educação) entre os anos de 2001 e 2012. Para essa pesquisa foi utilizado o método de mapeamento sistemático e método de pesquisa secundário que foi empregado para integrar os resultados oriundos de diversos estudos publicados anteriormente.

4. Metodologia

A pesquisa foi feita em 2 partes: pesquisa automatizada, onde foram utilizadas ferramentas de buscas; e pesquisa manual: onde periódicos de eventos foram verificados para complementar a busca. O enfoque da pesquisa é qualitativo, onde o pesquisador vai a campo buscando "captar" o fenômeno em estudo a partir da perspectiva das pessoas nele envolvidas, considerando todos os pontos de vista relevantes. Vários tipos de dados são coletados e analisados para que se entenda a dinâmica do fenômeno, Godoy (1995).

O mapeamento foi divido em 5 etapas. Primeira etapa: definição do tema e avaliação de critérios de inclusão e exclusão dos trabalhos a serem pesquisados na literatura, determinando a *string* de busca. Segunda etapa: leitura dos títulos e resumos

obtidos na literatura, pré-selecionando os estudos que atendem aos critérios estabelecidos anteriormente. Terceira etapa: leitura completa dos estudos pré-selecionados, reavaliando-os novamente, através dos critérios anteriormente definidos. Quarta etapa: análise das informações obtidas na terceira etapa. E finalmente na Quinta etapa: apresentação dos resultados obtidos.

5. Mapeamento Sistemático

Conceitualmente, Mapeamento Sistemático é projetado para prover uma visão mais ampla de um tópico de pesquisa, de modo a estabelecer se há evidência de pesquisa nesse tópico e prover uma indicação da quantidade de evidência, Kitchenham e Charters (2007).

5.1. Processo de busca

O processo de busca foi realizado automaticamente na ferramenta de pesquisa *web Google Scholar*, primeiramente sem incluir bases, e após a coleta dos resultados, foram realizadas novas pesquisas incluindo as bases de dados: IEEE, Elsevier, Springer. Uma pesquisa automática também foi realizada no portal de periódicos da Capes.

5.2 Questões de pesquisa

Para determinar o que está sendo pesquisado, algumas perguntas foram formuladas a fim de guiar o MS. Pergunta 1: Existe algum modelo, *framework*, protocolo ou simulador para desenvolver aplicações utilizando redes ubíquas? Pergunta 2: Que tipo de aplicações vem sendo desenvolvidas?

5.3 String de busca

A *string* de busca foi moldada de acordo com a ferramenta de pesquisa *web Google Scholar*, onde as palavras chaves utilizadas provém do assunto principal. A busca se apresenta da seguinte forma: i) Com todas as palavras: redes ubíquas. ii) Com no mínimo uma das palavras: "redes ubíquas" OR "*ubiquitous networks*" OR "redes sensíveis contexto" OR "*network context sensitive*" OR "redes pervasivas" OR "*pervasive network*". iii) Sem as palavras: "redes de sensores" - "*sensor networks*". iv) Período: 2010 – 2015.

Para realizar a busca avançada no portal de periódicos da Capes, foram pesquisadas duas palavras chaves por vez, totalizando três consultas: 1)"redes ubíquas" OR "*ubiquitous networks*"; 2) "redes sensíveis contexto" *OR* "*network context sensitive*" *OR* e 3) "redes pervasivas" e "*pervasive network*".

5.4 Critérios de inclusão e exclusão

A partir das questões formuladas, critérios de inclusão e exclusão foram definidos para refinar a pesquisa. O objetivo do estabelecimento destes critérios é impedir a sobrecarga de informações não relevantes para alcançar os resultados desejados.

Para a primeira etapa da pesquisa, estes critérios foram utilizados em forma de palavras chaves para incluir na ferramenta de busca. Inicialmente foram selecionadas as seguintes palavras chaves: "redes de sensores" - "*sensor networks*" - "*healthcare*" - "*homecare*" "redes sociais" - "*social networks*" - "redes sociais pervasivas" - "*pervasive*

social networks", mas devido a limitações da ferramenta de busca, estas palavras chaves foram reduzidas conforme a relevância: "redes de sensores" -"*sensor networks*".

A segunda etapa foi a fase de filtragem dos estudos pré-selecionados na etapa anterior, onde foram utilizados critérios mais específicos. Critérios de inclusão: i) Apresenta estudos e/ou aplicações com foco em redes ubíquas/pervasivas; ii) Apresenta estudos voltados a métodos, frameworks, protocolos entre outros que visam melhorias e inovações na área. E critérios de exclusão: i) Apresenta estudos e/ou aplicações com foco em redes ubíquas/pervasivas; ii) Estudos cujas aplicações são abordadas repetitivamente na literatura; e iii) Estudos que não são de livre acesso.

A pesquisa realizada na segunda etapa no *Google Scholar*, definindo a Springer como base de dados, retornou 195 publicações, onde foram selecionados 6 para a fase de extração, mas devido à restrição de acesso, os mesmos não foram incluídos, por não ser possível a visualização o texto completo para a retirada das informações relevantes sem a opção de compra.

Os estudos selecionados sem definição de base e das bases IEEE, Elsevier e Capes, como pode ser observado na Tabela 1, passaram para a terceira etapa onde foram relidos e reavaliados para serem aceitos ou não para avaliação dos resultados feito na quarta etapa.

Tabela 1. Resumo do Mapeamento Sistemático

Bases Eletrônicas	**Busca inicial**	**Primeira fase**	**Segunda fase**
		Incluídos	Incluídos
Sem base definida	1.680	26	11
IEEE	281	13	2
Elsevier	64	5	1
/Capes	314	7	3
Total	**2.339**	**51**	**17**

Já na quinta etapa, foram transcritas as informações relevantes dos estudos para análise e discussão dos resultados.

6. Resultados

Através dos critérios de busca foram selecionados os estudos que melhor representam o objetivo deste trabalho, nessa Seção serão apresentados um breve resumo e uma análise das informações coletadas.

A busca resultou em 23 artigos, onde 6 provenientes da base de dados Springer foram excluídos por não atenderem ao critério de livre acesso, restando então 17 artigos para serem analisados.

Para melhor compreender os 17 artigos selecionados para análise, os artigos foram organizados na conforme tabela 2.

Tabela 2 - Artigos selecionados para análise

CAPES		
1	*Editorial for Special Issue on "Challenges Pervasive Network and Applications for Internet of Things"*. Kyu Won Choi & Haiqing Nan	2014
2	*Intrusions Detection System Based on Ubiquitous Network Nodes.* Lynda Sellami et al	2014
3	*Smart Ubiquitous Networks for future telecommunication environments.* Chae Sub Lee et al	2014
4	*Standardization and Challenges of Smart Ubiquitous Networks in ITU-T.* Chae Sub Lee et al	2013
ELSEVIER		
5	*Combining data naming and context awareness for pervasive networks*. Paulo Mendes	2014
IEEE		
6	*A Method of Designing Seamless Connectivity Algorithm in Ubiquitous Networks.* Utsav Sinha	2010
7	*Content Delivery in Smart Ubiquitous Network.* Hongseok Jeon et al	2014
8	*Context Management for User-centric Context-aware Services over Pervasive Networks.* Sin-seok Seo et al	2012
9	*Managing Ubiquitous Networks – How do they do it?* Sven van der Meer et al	2010
10	*ubiSOAP: A Service-Oriented Middleware for Ubiquitous Networking.* Mauro Caporuscio et al	2012
SBD		
11	*Context Awareness for Smart Ubiquitous Networks*. Jeong Yun Kim e Gyu Myoung Lee	2013
12	*Method and System for Providing Context Awareness Based Networking Operation in Smart Ubiquitous Networks*. Jeong Yun Kim e Gyu Myoung Lee	2014
13	*PERORMANCE ANALYSIS OF A NODE MONITORING PROTOCOL IN UBIQUITOUS NETWORKS.* Sarada Prasad Gochhayat e Pallapa Venkataram	2010
14	*Providing Ubiquitous Networks Securely Using Host Identity Protocol (HIP).* Akihiro Takahashi e Yasuo Okabe	2011
15	*Seamless Wireless RSVP over Ubiquitous Networks.* Yu-Chang Chen et al	2011
16	*Secure Mobile Authentication in Ubiquitous Networking Environments*. Abdullah Mohammed A. Almuhaideb	2013
17	*System and a Method for Managing Device Identifier of A Ubiquitous Network*. Hui Li, Yu Li, Jue Jia, Changjun Zhao	2013

Com a leitura dos 17 artigos, houve uma percepção das principais abordagens sobre Redes Ubíquas, onde foi possível verificar e ideia principal de cada trabalho. Foi

plausível separar os artigos em temas centrais, ou seja, categorias. Cabe ressaltar que as categorias não são excludentes, alguns artigos abordam mais de um tema.

Os temas foram divididos em 7 categorias: i) Aplicabilidade de Redes Ubíquas; ii) Segurança em Redes Ubíquas; iii) Redes Ubíquas Inteligentes; iv) Estrutura para Redes Ubíquas; v) Protocolos/Algoritmos para conexão de Redes Ubíquas; vi) Arquitetura de Gerenciamento de Redes Ubíquas; e vii) Simuladores para Redes Ubíquas.

Classificados os artigos, observou-se que os estudos sobre Redes Ubíquas Inteligentes (*Smart Ubiquitous Networks – SUN*) é o mais frequente entre os trabalhos com 29,41%, seguidos por Protocolos/Algoritmos de conexão para Redes Ubíquas e Arquitetura de Gerenciamento com 17,65%.

Tabela 3 - Categorias dos artigos pesquisados

Temas dos Artigos	Nº dos Artigos	Porcentagem
Aplicabilidade de Redes Ubíquas	1	5,88%
Segurança em Redes Ubíquas	2, 16	11,76%
Redes Ubíquas Inteligentes	3, 4, 7, 11, 12	29,41%
Estrutura para Redes Ubíquas	5, 15	11,76%
Protocolos/Algoritmo de conexão para Redes Ubíquas	6, 13, 14	17,65%
Arquitetura de Gerenciamento	8, 10, 17	17,65%
Simuladores para Redes Ubíquas	9	5,88%

Com as informações contidas na Tabela 3, fica evidenciado como está os estudos e pesquisas sobre Redes Ubíquas dentro das parametrizações de inclusão e exclusão propostas nesse artigo.

7. Considerações Finais

Com o Mapeamento Sistemático realizado neste trabalho, puderam ser obtidas várias respostas sobre os últimos estudos que estão sendo pesquisados sobre Redes Ubíquas, principalmente os focos das pesquisas e também a motivação para a possibilidade de outros trabalhos correlacionados.

Nota-se que entre os 17 artigos selecionados não houve nenhum de pesquisa nacional, sendo assim, não foi possível informar como andam as pesquisas no Brasil sobre Redes Ubíquas. Acredita-se ser importante mensurar informações sobre os estudos em nossos meios acadêmicos. Nesse raciocínio sugere-se como trabalho futuro um MS de Redes Ubíquas só em trabalhos de nosso país.

Referências

Abreu, F., Almeida, A., Barreiros, E., & Saraiva, J. (2012). Métodos, Técnicas e Ferramentas para o Desenvolvimento de Software Educacional : Um Mapeamento

Sistemático . *Anais Do Simpósio Brasileiro de Informática Na Educação*, (Sbie), 26–30.

Atzori. L.; Iera. A.; Morabito. G. (2010). The Internet of things: a survey, Comput. Netw. 54 (15) 2787–2805.

Borges, S. D. S., Reis, H. M., Durelli, V. H. S., Bittencourt, I. I., Jaques, P. a., & Isotani, S. (2013). Gamificação Aplicada à Educação: Um Mapeamento Sistemático. *Anais Do Simpósio Brasileiro de Informática Educativa*, (Cbie), 234–243. http://doi.org/10.5753/CBIE.SBIE.2013.234.

Godoy, A. S. (1995). Pesquisa qualitativa: tipos fundamentais. *Revista de Administração de Empresas*, *35*(3), 20–29. http://doi.org/10.1590/S0034-75901995000300004.

Kitchenham, B.; Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE-2007-01, Keele University.

Magalhães, C. V., Santos, R. E., da Silva, F. Q., & Gomes, A. S. (2013). Caracterizando a pesquisa em informática na educação no Brasil: um mapeamento sistemático das publicações do SBIE. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)* (Vol. 24, No. 1, p. 22).

McCann, J. a., & Sterritt, R. (2010). Autonomic Pervasive Networks (APNs): Extended abstract. *Proceedings of the 7th IEEE International Conference and Workshop on Engineering of Autonomic and Autonomous Systems, EASe 2010*, 145–148. http://doi.org/10.1109/EASe.2010.27.

Obaidat, M. S., Denko, M., & Woungang, I. (2011). *Pervasive Computing and Networking*. *Pervasive Computing and Networking*. http://doi.org/10.1002/9781119970422.

Sofia. R.; Mendes. P.; Damasio. J.M.; Henriques.S.; Giglietto. F.; Giambitto. E ,et al. (2012) Moving towards a socially-driven internet architectural design. ACM CCR; 42(3).

Thabo K. R. Nkwe, Mieso K. Denko, and Jason B. Ernst. (2011). Pervasive Computing and Networking. Department of Computing and Information Science, University of Guelph, Guelph, Ontario,N1G 2W1, Canada. 221-236.

Weiser, M. The Computer for the 21st Century. Scientific American, v. 265, p. 94–104, 1991.

Yeun, C. Y. Y. C. Y., Lua, E. K. L. E. K., & Crowcroft, J. (2005). Security for emerging ubiquitous networks. *VTC-2005-Fall. 2005 IEEE 62nd Vehicular Technology Conference, 2005.*, *2*. http://doi.org/10.1109/VETECF.2005.1558125.

Algoritmo para sincronização de relógios físicos em sistemas distribuídos

Rônitti Juner da S. Rodrigues[1], Robson A. Lima[1], Diógenes Antonio M. José[1]

[1]Universidade do Estado de Mato Grosso - UNEMAT
Faculdade de Ciências Exatas e Tecnológicas - FACET
Coordenação de Ciência da Computação - CCC
Caixa Postal 92 - 78.390-000 - Barra do Bugres - MT - Brasil

{ronittijuner, robson.conq, dioxfile}@gmail.com

Abstract. *The synchronization of clocks in distributed systems allows the communication of processes in an orderly way, however its comprehension and execution are not a trivial task. Therefore, this article proposes an application capable of synchronizing physical clocks, considering Lamport's local time concept. The application was developed in Python and its evaluation was performed in a testbed with 16 computers. The results showed that the proposed application synchronized, in seconds, all computers clocks. In addition, it can be used as support in learning distributed systems since it allows to visualize the theoretical concepts of synchronization in practice.*

Resumo. *A sincronização de relógios em sistemas distribuídos permite a comunicação de processos de forma ordenada, todavia sua compreensão e execução não são uma tarefa trivial. Portanto, este artigo propõe um aplicativo capaz de sincronizar relógios físicos, considerando o conceito de hora local de Lamport. O aplicativo foi desenvolvido em Python e sua avaliação foi realizada em um testbed com 16 computadores. Os resultados mostraram que o aplicativo proposto sincronizou, em poucos segundos, todos os relógios dos computadores. Além disso, o mesmo pode ser usado como auxílio no aprendizado de sistemas distribuídos uma vez que ele possibilita visualizar na prática os conceitos teóricos de sincronização.*

1. Introdução

A sincronização de relógios em sistemas distribuídos consiste em manter os relógios dos computadores atualizados de modo que processos locais ou em diferentes computadores possam se comunicar de forma ordenada. Com o tempo, os temporizadores dos computadores, também conhecidos como relógios, sofrem uma defasagem. Neste contexto, é impossível garantir que todos os relógios, em todos os computadores, funcionem exatamente na mesma frequência [1]. Todavia, com um único computador e apenas um relógio não é necessário se preocupar com a defasagem do mesmo. Entretanto, em sistemas distribuídos, a defasagem dos relógios pode se tornar um sério problema. Por exemplo, o programa *make* do *Unix* é usado geralmente em projetos que possuem uma grande quantidade de códigos fontes a serem compilados. O funcionamento do programa é básico. Se o arquivo fonte *teste.c* foi modificado às *16:35* e o arquivo objeto *teste.o* marca a hora *16:30*, então o arquivo fonte *teste.c* foi alterado e precisa ser compilado. Dessa forma, o

make compilará apenas os arquivos fontes que foram modificados. Agora, imagine que a mesma situação aconteça em um sistema distribuído em que o fonte *teste.c* é editado em um computador com a seguinte marca de tempo *16:10*, já o arquivo objeto está em um computador remoto com a seguinte marca de tempo *16:50*. Assim, o *make* não compilará o código alterado e o arquivo objeto não funcionará como deveria.

Há muitos outros exemplos em que é necessário ter um horário global comum. Entretanto, o que importa é saber qual evento aconteceu antes do outro. No caso do *make*, basta apenas saber qual arquivo fonte foi alterado antes do arquivo objeto ser gerado. Há varias décadas cientistas desenvolvem algoritmos para a sincronização de relógios [2, 3, 4]. A sincronização é importante para vários tipos de aplicações, principalmente as que funcionam em sistemas distribuídos.

Diante do exposto, o objetivo deste trabalho consiste em desenvolver um aplicativo para realizar a sincronização de relógios utilizando o conceito de hora local desenvolvido por Lamport [2]. Além disso, como objetivo secundário a proposta de aplicativo visa auxiliar o processo de aprendizagem de sincronização de relógios na disciplina de sistemas distribuídos, uma vez que o aplicativo em questão permitirá visualizar na prática os conceitos que são vistos apenas em teoria. O aplicativo foi desenvolvido em *Python* e o mesmo foi avaliado em um *testbed* com 16 computadores no laboratório de redes da Universidade do Estado de Mato Grosso *(UNEMAT)*, Campus de Barra do Bugres. A solução proposta foi avaliada em sistemas *Unix*, todavia a mesma pode ser instalada em sistemas *Microsoft Windows*. Os resultados mostraram que a aplicação mantém todos os computadores sincronizados por meio da difusão de suas horas locais.

Para melhor compreensão do conteúdo, o artigo foi organizado como segue: a Seção 2 apresenta os trabalhos relacionados. Na Seção 3, é apresentada a metodologia utilizada. A Seção 4 descreve, em detalhes, o aplicativo proposto. Na Seção 5, são apresentados os resultados e discussão. E por fim, na Seção 6, são apresentadas a conclusão e as possibilidades de trabalhos futuros.

2. Trabalhos relacionados

O objetivo desta seção consiste em descrever alguns dos principais trabalhos sobre sincronização de relógios, os quais servem como base para compreensão do aplicativo proposto.

No trabalho de Cristian *et al.* [4] é proposto um protocolo prático para a sincronização de relógios em ambientes distribuídos. A importância do desenvolvimento desse protocolo se deve ao fato da implementação de algoritmos que funcionam na presença de omissões, por exemplo, falhas do processador e/ou enlaces com auto atraso. Assim, o tempo é mantido sincronizado dentro de uma faixa de erro previamente definida e, dessa forma, é possível saber o tempo decorrido entre eventos de um mesmo processo. Por exemplo, um cliente solicita o tempo de um servidor. Após receber a solicitação o servidor envia uma resposta ao cliente anexando o valor de seu relógio T. Então, o cliente configura o tempo calculando $T + Round\ Trip\ Time\ (RTT)/2$.

Lamport em [2] trata da sincronização de eventos que ocorrem em processos locais e remotos levando em consideração a hora local. Segundo Lamport, a hora global não é importante, mas sim a relação *aconteceu antes*. A relação *aconteceu antes* é composta por três condições básicas: *(I)* Se a e b são eventos em um mesmo processo e a acontece

antes de b, então $a \rightarrow b$; *(II)* Se a é o envio de uma mensagem por um processo e b é o recebimento da mesma mensagem por outro processo, então $a \rightarrow b$; e *(III)* Se $a \rightarrow b$ e $b \rightarrow c$, então $a \rightarrow c$. Neste mesmo trabalho, também é definido o conceito de hora local através de relógios lógicos. Um relógio lógico é um número associado a um evento. Tal número representa o tempo no qual o evento aconteceu. A representação é feita da seguinte forma: cada processo P_i possui uma função C_i que determina um número $C_i(a)$ para cada evento a naquele processo. Assim, todo o processo é representado por meio da função C, que assimila a qualquer evento b o número $C(b)$, onde $C(b) = C_j(b)$ se b é um evento no processo P_j. Embora C_i seja uma representação lógica que não tem nenhuma relação com os relógios físicos, este conceito pode ser aplicado em redes que não estão conectadas à *Internet* ou que não possuem um servidor de hora local, pois estas podem usar o conceito de hora local para sincronizarem seus relógios.

Guzella e Zatti [5] propõem um algoritmo de sincronização denominado *Algoritmo de Berkeley*. Este algoritmo é ativo e consulta o tempo através de um *daemon*. Um servidor rodando um *daemon* de tempo pergunta aos outros computadores as horas de seus relógios, estes respondem, então o servidor calcula a hora média e informa a cada computador como deve acertar seu relógio.

O *Network Time Protocol (NTP) RFC 1059*, idealizado por David Mills da Universidade de Delaware (*USA*), consiste em um protocolo para sincronização de relógios dos computadores. O *NTP* usa o protocolo *User Datagram Protocol (UDP)* para receber solicitações na porta 123. Atualmente o *NTP* está na versão 4 *RFC 5905* e é amplamente utilizado para sincronizar computadores na *Internet* através de servidores de hora. Sua estrutura é composta por uma hierarquia em árvore e tem como fonte de referência a estação de rádio do *national standard time* [6]. Ele pode ser instalado no sistema operacional, *Windows* ou *Linux*, entretanto é necessário ter como base uma hora de referência para seu funcionamento.

3. Descrição da proposta

O aplicativo foi desenvolvido em linguagem *Python* com característica *multithreading* e utilizando programação orientada a objetos. Sua arquitetura é composta de três classes: *TSGUI*, *TSServer* e *TSClient*. A classe *TSGUI* é responsável por gerar e manipular a interface gráfica, que é apresentada ao usuário. Nesta interface, Figura 1, é possível fazer o controle de início da sincronização, visualizar hora atual e verificar quais computadores estão em sincronia. Ela faz uso da biblioteca *PyGTK 2.26*, disponível sob licença *LGPL*, que permite facilmente criar os componentes como menus, controles e *widgets* utilizados na interface.

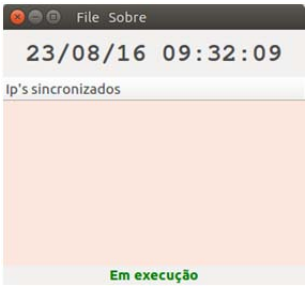


Figura 1. Interface gráfica do aplicativo.

A classe *TSServer* implementa as funcionalidades do servidor. Este é responsável por receber e analisar as mensagens originadas do cliente. Esta instância é executada em uma *Thread* e se mantém aguardando uma mensagem. Assim, quando o servidor recebe uma mensagem com o *timestamp* do relógio do cliente, ele compara o tempo recebido com o seu e, caso seja menor, envia por meio de uma mensagem *unicast* o seu relógio local para o cliente (e.g., um valor float em segundos e milissegundos). Se o valor recebido for igual, maior, ou originar do mesmo computador em que o servidor está executando, nenhuma resposta é retornada pelo servidor. O Algoritmo 1 apresenta as ações executadas no servidor.

Algorithm 1 Pseudocódigo da função *listen* da classe servidor.

```
Input: Relogio Remoto
Output: Relogio Local
1: procedure LISTEN
2:     while true do
3:         relogioremoto, ipcliente ← SOCK.RECVFROM()
4:         if relogioremoto < relogiolocal then
5:             SOCK.SENDTO(ipremoto , relogiolocal)
6:         end if
7:     end while
8: end procedure
```

A classe *TSCliente* implementa as funcionalidades do cliente. Este difunde uma mensagem com seu tempo atual, em segundos e milissegundos, para toda a rede (e.g., *broadcast*). A partir deste momento, o cliente aguarda o recebimento de uma mensagem de algum servidor. Ele aguarda por um tempo máximo de $1s$ e se nenhuma mensagem for recebida, uma exceção é gerada por estouro do *timeout*, e a espera da mensagem é interrompida. Todavia, quando uma mensagem é recebida, faz-se uma estimativa simples do atraso que possibilita a sincronização do relógio local, Algoritmo 2. Basicamente

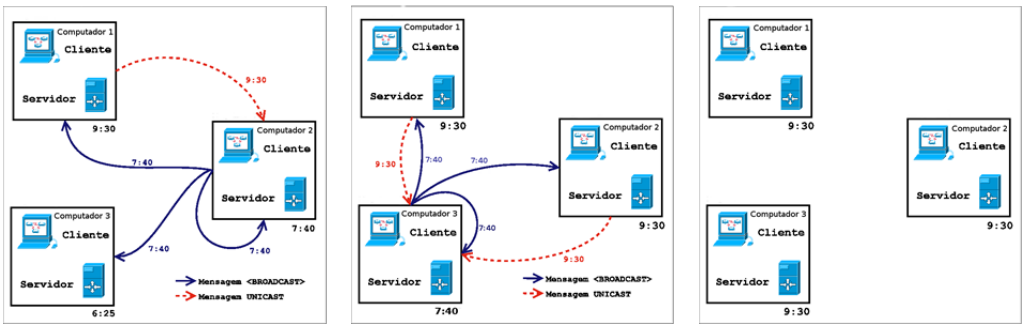
Algorithm 2 Pseudocódigo da função *sendmessage* da classe cliente.

```
Input: Relogio Remoto
Output: Relogio Local
 1: procedure SENDMESSAGE
 2:     SOCK.SENDTO(broadcast, relogiolocal)
 3:     datadeenvio ← relogiolocal
 4:     relogioremoto , ipservidor ← SOCK.RECVFROM( )
 5:     if relogioremoto then
 6:         atraso ← (datadeenvio − relogiolocal) ÷ 2
 7:         if relogioremoto > relogiolocal then
 8:             relogioremoto ← relogioremoto + atraso
 9:             CHANGETIME(relogioremoto)
10:         end if
11:     end if
12: end procedure
```

o tempo de atraso T_{atraso} do Algoritmo 2 é calculado com base no método de Cristian apresentado em [4]. O tempo T_{envio} é registrado no momento do envio da mensagem e quando uma mensagem é retornada ao cliente obtém-se $T_{recebido}$. Tem-se então o tempo $T_{RTT} = T_{recebido} - T_{envio}$. Assumindo que o tempo é igualmente dividido, obtém-se: $T_{atraso} = T_{RTT}/2$. Com T_{atraso} obtido, então é feita a soma com o tempo recebido do servidor T_{serv} e esta é comparada com a hora atual do cliente. Se a hora do cliente for

menor, o relógio do cliente é configurado com $T_{serv} + T_{atraso}$. Para manter a sincronia, o cliente envia a cada $5s$ uma mensagem na rede.

Por exemplo, considere três computadores interligados por uma rede, como mostra a Figura 2. Eles têm seus relógios fora de sincronia. Note que na Figura 2(a), o cliente no computador 2 envia uma mensagem com o *timestamp*, por *broadcast*, na rede. Observa-se também que os computadores 1, 2 e 3 recebem a mesma mensagem. Entretanto, somente o computador 1 possui seu relógio maior, adiantado, que o da mensagem recebida. Assim, somente o servidor deste computador envia ao computador 2 uma mensagem *unicast* contendo seu relógio. Deste modo, o computador 2 atualiza seu relógio com o valor recebido. Na Figura 2(b), o cliente do computador 3 envia uma mensagem via *broadcast* e os computadores 1 e 2 recebem a mesma. Os servidores dos dois computadores verificam que o valor de relógio recebido é menor. Dessa forma, cada um envia uma mensagem com o valor de seu relógio para o computador 3. A primeira mensagem que chega até ele é recebida, e seu relógio é atualizado. Assim, observa-se na Figura 2(c) que os relógios de todos os computadores foram sincronizados.



(a) Computador 2 sincronizando. (b) Computador 3 sincronizando. (c) Computadores sincronizados.

Figura 2. Modelo de sincronização com três computadores.

4. Materiais e métodos

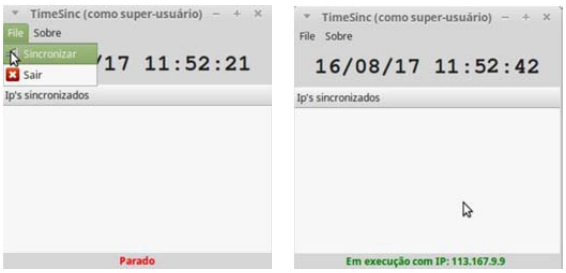
A técnica de pesquisa utilizada neste trabalho foi a revisão bibliográfica, tendo como base referenciais teóricos que tratam de sincronização como o apresentado por Lamport. Portanto, para o desenvolvimento do aplicativo e execução dos testes, a metodologia foi dada por:

- **Softwares utilizados**: linguagem de programação *Python 2.7*, biblioteca de interface gráfica *PyGTK 2.26*, IDE *Eclipse 4.6.0 (Neon)* com o *plugin PyDev 5.1*, Sistema Operacional *Ubuntu 16.04 LTS 64bits*, e Sistema Operacional *Linux Mint 17 32bits*;
- **Tipo de socket**: *socket UDP*, pois as mensagens são trocadas por *broadcast* e isso não seria possível em um *socket* orientado à conexão como o *TCP*;
- **Hardware para desenvolvimento**: *Intel (R) Core (TM) i5-3337U CPU 1.80GHz ×4*, *8GB* de memoria *RAM*, *HD* híbrido *500GB + 32GB SSD*;
- **Hardware do ambiente de teste**: *Intel (R) Pentium 4 (TM) 3.0GHz*, *1GB* de memoria *RAM*, *HD 40 GB*, rede *Fast Ethernet 10/100*;
- **Testes realizados**: os testes do aplicativo foram feitos em um *testbed* com 16 computadores, *Laboratório de Redes de Computadores da (UNEMAT)*, e todos os computadores do laboratório foram configurados em uma rede classe

A 113.167.9.0/24. Para que a sincronização ocorre-se, foi adotado como hora correta a maior hora. Dessa forma, os relógios de 15 computadores foram atrasados e o 16º foi configurado com a maior hora local. Além disso, foi verificado se todos os computadores atualizavam corretamente. Entre os testes realizados, um dos computadores teve sua hora adiantada para perceber se todos os outros também seriam atualizados, além disso, cada computador foi atrasado manualmente para verificar se ele entrava novamente em sincronia.

5. Resultados e Discussão

Para efeitos de simplificação, esta seção apresenta os resultados dos testes realizados em apenas três computadores. Os testes de sincronização foram feitos nos computadores 113.167.9.9, 113.167.9.13 e 113.167.9.14. O primeiro passo consistiu em executar o aplicativo, para isso foi necessário instalar a biblioteca *PyGTK* na versão 2.26. Como é necessário alterar a hora do sistema, o aplicativo foi iniciado como administrador, *root*. Ao ser iniciado, o aplicativo apresenta a data e hora corrente e a sincronização é acionada através do menu *"file"*, Figura 3.

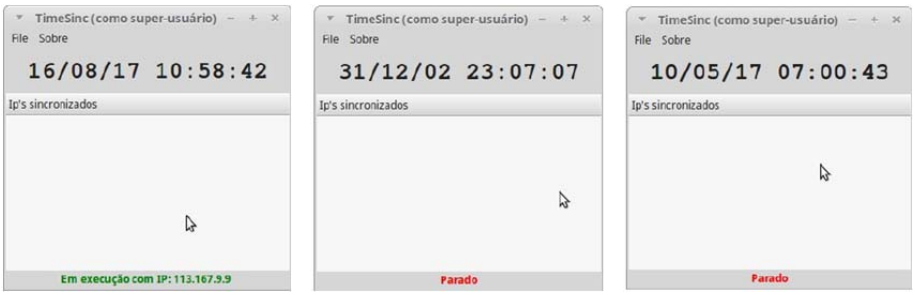


(a) Tela 1, parado. (b) Tela 2, em execução.

Figura 3. Iniciando a sincronização no computador 113.167.9.9.

A interface gráfica desenvolvida para o aplicativo é simples, intuitiva e de fácil uso. Nela é apresentado o relógio local com a data e hora, eliminando assim a necessidade de verificar a hora na bandeja do sistema. O aplicativo foi iniciado em três computadores, de modo que o computador 1 teve sua hora correta mantida e os computadores 2 e 3 tiveram suas horas atrasadas, Figura 4(a)(b)(c). Conforme mostra a Figura 4(d)(e), os computadores 2 e 3 foram atualizados com base no computador 1. Observa-se que há um espaço de tempo entre as horas, em função do tempo que foi levado para logar em cada computador e executar o aplicativo em cada um deles. Observa-se também que o aplicativo no computador 3, Figura 4(e), foi executado antes de sua execução no computador 2, Figura 4(d). Por causa disso, existe uma diferença de tempo nas horas apresentadas: *10:48:42* (computador 1), *11:10:32* (computador 3) e *11:14:38* (computador 2).

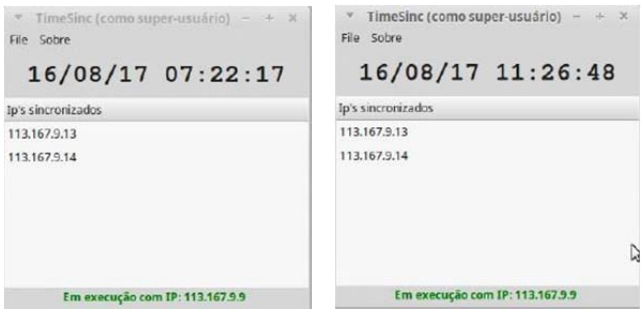
A Figura 5(a) apresenta outro teste no qual a hora do computador 1 é atrasada. Logo após o atraso, ele entrou novamente em sincronia com os computadores 2 e 3, Figura 5(b). Em outro teste, o computador 3 teve a hora adiantada, Figura 6(a). Assim, todos os outros computadores tiveram suas horas sincronizadas com o mesmo, por exemplo, o computador 2, Figura 6(b). A aplicação também gera dois arquivos de *log* (e.g., *server.log* e *client.log*), que possibilitam analisar as mensagens enviadas e recebidas por cada entidade, Figura 7.



(a) Computador 1. (b) Computador 2 atrasado. (c) Computador 3 atrasado.

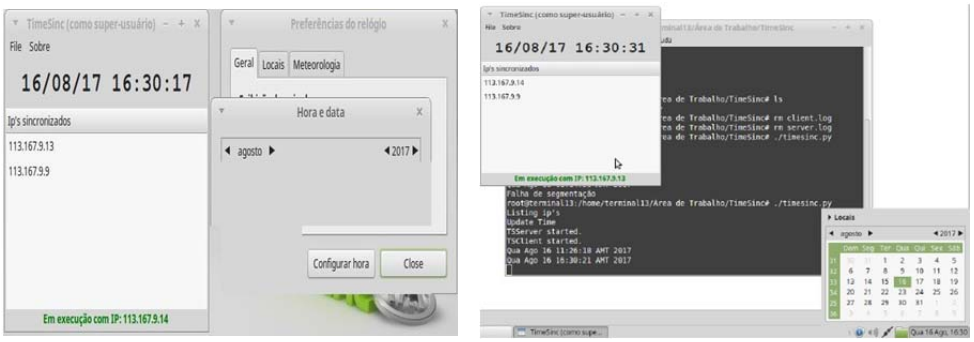
(d) Computador 2 sincronizado. (e) Computador 3 sincronizado.

Figura 4. Configuração da sincronização em três computadores.



(a) Computador 1 atrasado. (b) Computador 1 sincronizado.

Figura 5. Atraso da hora no computador 1.



(a) Computador 3, hora adiantada. (b) Computador 2 sincronizado.

Figura 6. Computador 3 com a hora adiantada para as 16:30.

```
Server: 113.167.9.14 is delayed. I'm answering.
Server: 113.167.9.14 sent to me 1502896240.8533, mine is 1502896240.8156
```

(a) Server *log*.

```
Client: Sending my time. 1502896364.0157
Response:  ServerTime 1502896235.9353 - MyTime 1502896364.0177 - Delay0.0010
```

(b) Client *log*.

Figura 7. Arquivos de *log* das entidades: servidor e cliente.

6. Conclusão

Este trabalho apresentou um aplicativo baseado no conceito de hora local proposto por Lamport, para sincronização de relógios físicos de *n* computadores ligados em rede. Além disso, o aplicativo proposto tem como um de seus objetivos servir como auxílio no aprendizado de sistemas distribuídos no que diz respeito à sincronização de relógios. Dentre as contribuições deste trabalho destacam-se: a proposta de um algoritmo de sincronização de relógios físicos em computadores legados; o desenvolvimento de um aplicativo em *Python* para sincronização de relógios; a utilização de programação *multithreading* e orientada a objetos; a aplicação dos conceitos de sincronização em sistemas distribuídos na resolução de um problema do mundo real; a utilização e promoção de *Software Livre*, pois todas as ferramentas usadas são livres; e o desenvolvimento de atividade prática e de laboratório como um exercício didático da disciplina de sistemas distribuídos.

Dessa forma, concluiu-se que o aplicativo proposto atingiu o objetivo esperado que consiste na sincronização de relógios em redes de computadores. Como trabalhos futuros, pode-se melhorar a eficiência do envio de mensagens, diminuindo o tráfego das mesmas na rede e consequentemente a economia de energia no caso de ambientes *wireless*. Entre as diversas possibilidades é possível implementar soluções tolerantes a falhas, por exemplo, falhas bizantinas, nós faltosos, queda de processos. Também, pode ser proposto a ordenação total dos eventos no aplicativo utilizando relógios vetoriais.

Referências

[1] A. S. Tanenbaum and M. V. Steen, *Sistemas Distribuídos: princípios e paradigmas*. São Paulo: Pearson Prentice Hall, 2th ed., 2007.

[2] L. Lamport, “Time, clocks, and the ordering of events in a distributed system,” *Communications of the ACM*, vol. 21, pp. 558–565, Jul 1978.

[3] L. Lamport and P. M. Melliar-Smith, “Synchronizing clocks in the presence of faults,” *Journal of the Association for Computing Machinery*, vol. 32, p. 27, January 1985.

[4] F. Cristian, H. Aghili, and R. Strong, “Clock synchronization in the presence of omission and perfomance failures, and processor joins,” in *16th IEEE Int. Symp. on Fault-tolerant Computing System*, (Vienna), 1986.

[5] R. Gusella and S. Zatti, “The accuracy of the clock synchronization achieved by tempo in berkeley unix 4.3bsd,” *IEEE Transactions on Software Engineering*, vol. 15, pp. 847–853, Jul 1989.

[6] D. Mill, “Network Time Protocol version 1.” `https://tools.ietf.org/html/rfc1059`, 1989. [Último acesso: 18-Agosto-2017].

Sistema de Máquinas Virtuais Windows® utilizando XenServer e OpenStack

Clóvisson L. Rosa[1] , Otávio P. Bragagnollo[1], Walther F. Pedrozo[1] , Tiago A. Rizzetti[1]

[1]Colégio Técnico Industrial de Santa Maria - Universidade Federal de Santa Maria (UFSM)
Av. Roraima no 1000 – 97.105-900 – Santa Maria - RS - Brasil

{clovisson.lopes,otavio_prestes,waltherpedrozo}@redes.ufsm.br ,
rizzetti@gmail.com

Abstract. *This paper presents the Windows® virtual machine system will be implemented at CTISM, it has the goal to provide virtual machines with Windows® operational system to all users in the institution with high performance. This project shows in the summarized and direct way the informations about the infrastructure, resources, features and optimization, besides future researches.*

Resumo. *Este artigo apresenta o sistema de máquinas virtuais Windows® que será implementado no CTISM, que tem como objetivo proporcionar máquinas virtuais com o sistema operacional Windows® de alto desempenho para os usuários da instituição. O trabalho mostra de forma resumida e objetiva as informações sobre a infraestrutura, recursos, funcionalidades e ainda melhorias, assim como trabalhos futuros.*

1. Introdução

O Colégio Técnico Industrial de Santa Maria (CTISM), atualmente possui um sistema de máquinas virtuais, conhecido como VMS, que permite a criação de máquinas virtuais por alunos, professores ou servidores, desde que possuam cadastro na instituição.

O sistema vem sendo utilizado principalmente como ferramenta de ensino durante as aulas e em projetos de pesquisa, como máquina de testes, servidores, etc.

O VMS possibilita a criação de múltiplas máquinas, onde o usuário tem permissão de administrador sobre-as, entretanto, o atual sistema trabalha apenas com distribuições Linux, impossibilitando o uso de certos softwares, que em alguns casos, estão disponíveis somente para plataforma Windows®.

No momento, a solução implantada é a criação de máquinas virtuais pelo VirtualBox, virtualizador desenvolvido pela Oracle, porém, o desempenho das máquinas criadas, é claramente inferior ao de uma máquina física. Como solução para esse problema, está em desenvolvimento um sistema que permita os usuários criarem máquinas Windows®, através de uma página Web.

O sistema utiliza o virtualizador Citrix XenServer, plataforma desenvolvida pela Citrix, que permite a criação de máquinas com desempenho quase que equivalente às de máquinas físicas. Isso se dá principalmente, pois ele trabalha com a técnica de virtualização total, tema que será abordado logo adiante. O sistema também fará uso do

OpenStack, plataforma de computação em nuvem *open source*, que permite criar nuvens privadas e públicas.

Atualmente, o sistema encontra-se com apenas o Citrix XenServer em funcionamento na infraestrutura do CTISM, entretanto, o OpenStack será implantado futuramente para trabalhar em conjunto com o XenServer.

2. Virtualização

Virtualização é uma técnica que permite distribuir e utilizar recursos (memória RAM, processamento, espaço em disco, etc.) entre um sistema operacional e outro, esse sendo chamado máquina virtual [NandaChiueh, 2005].

Uma máquina virtual pode ser definida como "uma duplicata eficiente e isolada de uma máquina real". A IBM define uma máquina virtual como uma cópia isolada de um sistema físico, e essa cópia está totalmente protegida [Laureano, 2006]. Assim, cada máquina virtual, então, possui seu próprio sistema operacional, aplicativos e serviços de rede [Carissimi, 2008].

Existem várias formas de virtualização, onde as mais conhecidas são: virtualização total e a paravirtualização.

2.1. Virtualização Total

A virtualização total fornece uma cópia do hardware do hospedeiro. Assim, o sistema operacional e os programas são executados na máquina virtual da mesma forma que seriam em uma máquina física [Carissimi, 2008].

Essa técnica possui uma grande vantagem, o sistema operacional não necessita ser modificado para ser virtualizado, porém, existe uma desvantagem, que é a perda de desempenho.

2.2. Paravirtualização

Ao contrário da virtualização total, na paravirtualização o sistema hóspede é modificado para que saiba que está sendo virtualizado, desse modo fazendo que o hypervisor não necessite testar instrução por instrução, somente as que possam alterar o estado sistema. Isto normalmente envolve a substituição de quaisquer operações privilegiadas, que só seriam executadas no Kernel, por chamadas para o hypervisor, conhecidas como hiperchamadas Essa solução aumenta a performance das máquinas virtuais que a utilizam [Barham, 2003].

Na paravirtualização é dispensada a utilização de drivers genéricos, já que, os dispositivos de hardware utilizarão drivers da própria máquina virtual, assim, utilizando sua capacidade total. Este tipo de virtualização tem um ganho de desempenho significativo frente a virtualização completa [Mattos, 2008].

Mesmo que a paravirtualização demonstre um melhor desempenho comparado a virtualização total, essa diferença vem sendo superada devido às instruções de virtualização nos processadores Intel (Intel VT), e AMD (AMD-V), favorecendo a virtualização total [Mattos, 2008].

Na Figura 1 é possível observar a representação desses dois tipos de virtualização.

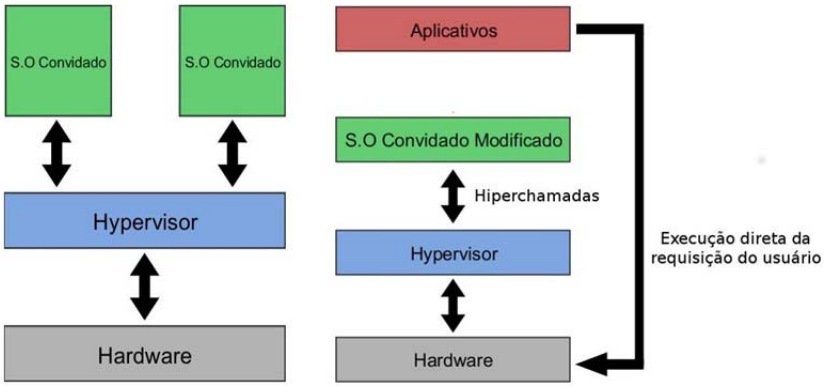


Figura 1. Virtualização total / Paravirtualização

3. Computação em Nuvem

A computação em nuvem consiste na computação como serviço em vez de um produto, podendo distribuir recursos computacionais, softwares e informações de forma dinâmica e sob demanda. Funciona de modo semelhante a serviços como água, energia elétrica ou telefone, onde o cliente é tarifado conforme o uso do serviço [Foster 2008].

Atualmente existem três tipos de modelos de serviços em Computação em Nuvem, organizados conforme o nível de gerência contratado pelo usuário. Esses são:

- Software como um Serviço (SaaS): oferece ao usuário acesso a uma aplicação. Informações sobre infraestrutura como sistema operacional, rede e armazenamento ficam invisíveis ao usuário. Exemplo: Google Drive, Microsoft Office 365.
- Plataforma como um Serviço (PaaS): oferece ao usuário uma plataforma de recursos gerenciável para dar suporte a aplicações. Nesse caso informações sobre infraestrutura também ficam invisíveis ao usuário.
- Infraestrutura como um serviço (IaaS): oferece ao usuário um conjunto de recursos computacionais, como capacidade de processamento, armazenamento e redes. Caso o usuário necessite, a quantidade de recursos pode ser aumentada.

3.1. Modelos de implantação de Computação em Nuvem

Atualmente existem três modelos de infraestrutura de rede para acesso aos dados na computação em nuvem. Eles são:

- Nuvem Pública: servidores compartilhados e acessados pela rede. Este tipo de nuvem é recomendada para startups e micro e pequenas empresas, devido ao seu baixo custo. Sua implantação é simples e rápida, uma vez que os servidores são alocados em data centers externos.
- Nuvem Privada: servidores em uma estrutura local. Este tipo de nuvem exige um pouco mais de investimento, pois é recomendada para as empresas de médio e grande porte, já que trabalha com um grande volume de demandas, além de contar com um tempo de resposta bem veloz.
- Nuvem híbrida: combinação das duas primeiras, utilizando as características de uma ou de outra quando é mais conveniente. Este tipo de nuvem permite a empresa armazenar dados locais e sigilosos em uma nuvem privada, e fazer a transferência deles entre ambas as nuvens, em teoria, a nuvem híbrida seria o modelo ideal para todas as empresas, pois a quantidade de recursos que ela oferece é extensa.

4. Trabalhos Relacionados

A virtualização e a computação em nuvem vem ganhando cada vez mais força, sendo assim, é possível encontrar algumas propostas semelhantes com diferentes fins.

Entretanto nenhuma proposta se assemelha tanto ao sistemas de máquinas virtuais Windows® do CTISM quanto a Nuvem UFRGS: Iass como ferramenta de apoio à pesquisa, que proporciona máquinas virtuais como ferramenta de apoio aos pesquisadores da instituição [Ribeiro, R. Q., Foscarini, E.D., 2014].

O Sistema de Máquinas Virtuais Windows®, diferencia-se do citado anteriormente por utilizar ferramentas atuais, com elevado número de usuários provendo um serviço de suporte de alta qualidade bem como, proporcionar esse sistema a todos os membros do CTISM, diferentemente da Nuvem UFRGS, que somente pesquisadores da instituição podem usufruir de tal serviço.

5. Sistemas da Máquinas Virtuais Windows®

O sistema de máquinas virtuais vem sendo desenvolvido desde o início do ano de 2017 e tem como objetivo proporcionar aos alunos, professores e servidores, uma maneira de virtualizar o Windows® de forma superior ao do VirtualBox.

Esse sistema permitirá que os usuários criem um número de máquinas limitado, através de uma página Web. O sistema oferecerá padrões de configurações para as máquinas, como disponibilidade de memória RAM, quantidade de núcleos do processador e armazenamento.

Dessa forma, a plataforma de virtualização adotada foi o XenServer, hypervisor de máquinas virtuais que é executado sobre o hardware (bare-metal) desenvolvido pela Citrix. Em conjunto com o XenServer será utilizado o OpenStack, software que será responsável pela distribuições de recursos computacionais para cada máquina virtual.

5.1. Ferramentas

O Citrix XenServer foi escolhido para ser o virtualizador do sistema de máquinas virtuais Windows® principalmente pelo fato de não haver custo de licenciamento e ter seu código fonte aberto [XenServer 1999-2017]. Além disso, está entre os melhores virtualizadores da atualidade, destacando-se por trabalhar de forma eficiente com os dois principais tipos de virtualização, a virtualização total e a paravirtualização, explicadas anteriormente.

O OpenStack é uma colaboração global entre desenvolvedores na área de computação em nuvem, que visa oferecer soluções para nuvens públicas e privadas. Este software é um conjunto de projetos *open sources* utilizados para configurar e operar infraestrutura e armazenamento em nuvem. Para a implementação do OpenStack, será utilizado o Devstack, conjunto de *scripts* que permitirão criar um ambiente completo do OpenStack.

5.2. Infraestrutura

O sistemas da máquinas virtuais Windows® foi criado com base nos recursos disponibilizados pelo CTISM e é composta pelos seguintes equipamento:

Tabela 1. Equipamentos

	Servidor 1	Servidor 2	Servidor 2
RAM	64 GB	12 GB	12 GB
Processador	Intel(R) Xeon(R) CPU E5-2640 v4 @ 2.40GHz	Intel(R) Xeon(R) CPU E5506,@ 2.13GHz	Intel(R) Xeon(R) CPU E5506,@ 2.13GHz
Armazenamento	5 TB	450 GB	450 GB

6. Resultados

Para analisar o desempenho do novo virtualizador, o Citrix XenServer, foram realizados diversos testes em tal. Esses testes se basearam em analisar o tempo que o virtualizador leva para realizar algumas ações.

As ações escolhidas para serem analisadas foram, tempo de inicialização, exclusão e migração de máquinas virtuais entre servidores, por serem fatores comuns entre virtualizadores, onde é possível visualizar a diferença de desempenho de forma clara.

Para a realização dos teste, todas as máquinas virtuais foram criadas com as mesmas configurações, Windows® 7 32-Bits, 2 GB de memória RAM, 30 GB de armazenamento e processador Intel com dois núcleos de 2.13 GHz.

Na Figura 2, é possível verificar o tempo de inicialização das máquinas virtuais em relação ao número de máquinas. Pode ser observado através deste que o tempo para inicialização das máquinas mostrou-se linear com o incremento do número de máquinas a serem inicializadas.

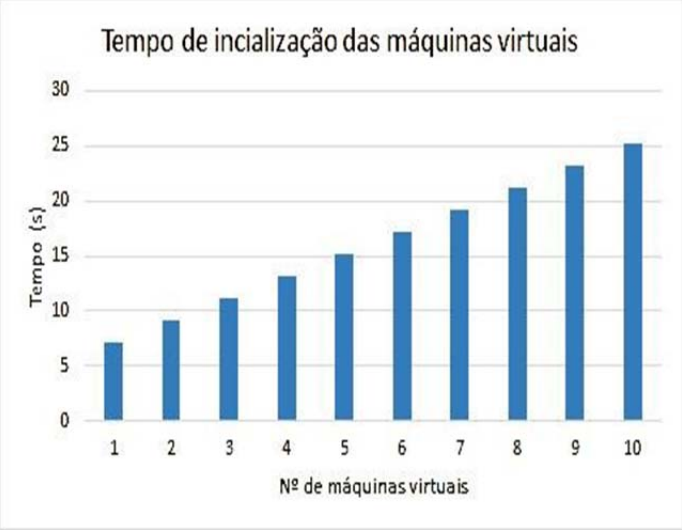


Figura 2. Gráfico da relação entre tempo x número de máquinas, para inicialização de máquinas

No gráfico representado na Figura 3, é expresso a relação de tempo em função do número de máquinas virtuais a serem migradas a outro servidor. A migração de máquinas virtuais entre servidores, traz ao sistema diversas vantagens, por exemplo, balanceamento de carga, que permite distribuir a carga de trabalho uniformemente entre servidores, função não encontrada no VirtualBox. É importante salientar que foram utilizados três servidores com o XenServer para realizar esse teste.

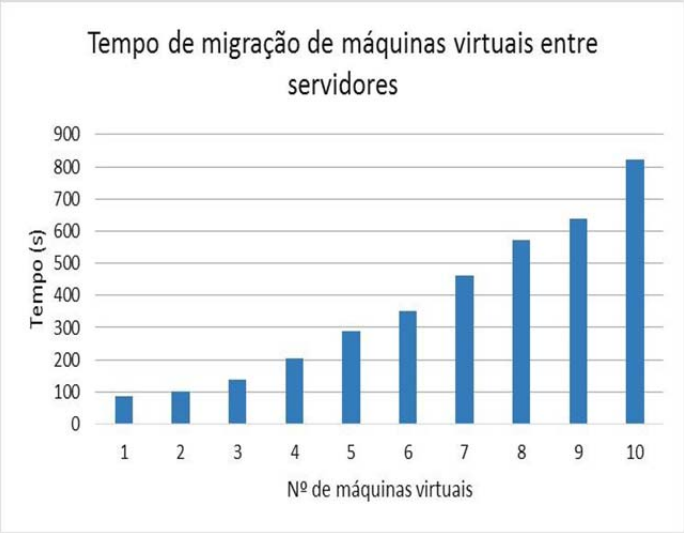


Figura 3. Gráfico da relação entre tempo x número de máquinas, para migração de máquinas

Como pode ser visto na Figura 4, é possível analisar o tempo de exclusão de determinados números de máquinas, lembrando, que todas as máquinas virtuais possuíam as mesmas configurações.

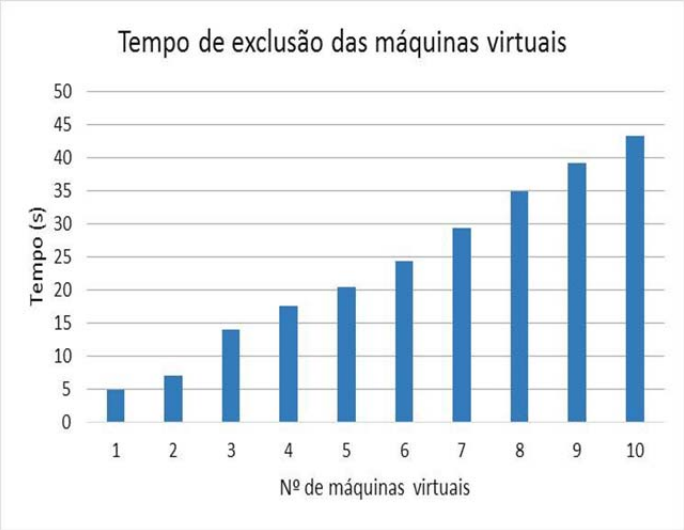


Figura 4. Gráfico da relação entre tempo x número de máquinas, para exclusão de máquinas.

7. Considerações Finais

O sistema da máquinas virtuais apresentado nesse trabalho sempre foi uma necessidade dos alunos, professores e servidores do CTISM. A criação do sistema de virtualização de máquinas virtuais Windows®, permitirá que os usuários criem máquinas virtuais de alto desempenho, auxiliando os usuários nos seus projetos e trabalhos.

Todos os testes apresentados mostraram resultados satisfatórios, como por exemplo, a possibilidade de migrar máquinas virtuais entre servidores de forma fácil,rápida,

dinâmica e segura.

Portanto, a principal contribuição desse artigo, é provar que é possível criar um sistema de virtualização eficiente e seguro com ferramentas *open sources*.

A próxima etapa a ser realizada, em trabalhos futuros, é implementar de forma definitiva o OpenStack, para que possa trabalhar em conjunto com o XenServer, para que finalmente, o sistema possa ser disponibilizado aos usuários do CTISM.

Referências

Barham, P. et al. (2003). Xen and the Art of Virtualization. http://www.cl.cam.ac.uk/research/srg/netos/papers/2003-xensosp.pdf

Carissimi, A. (2008). Virtualização: da teoria a soluções. Minicursos do Simpósio Brasileiro de Redes de Computadores (SBRC).

Foster, I. et al.(2008). Cloud computing and grid computing 360-degree compared. Grid Computing Environments Workshop, 2008. GCE'08. Ieee, 2008.p.1-10.

GTA/COPPE/UFRJ- Mestrado e Doutorado em Redes de Computadores. Virtualização total e para-virtualização https://www.gta.ufrj.br/grad/08 _ 1/virtual/Virtualizaototalepara-virtualizao.html, July.

Laureano, M. (2006). Máquinas virtuais e emuladores: conceitos, técnicas e aplicações São Paulo: Novatec

Mattos, D. M. F. (2008). Virtualização: VMWare e Xen. Grupo de Teleinformática e Automação da UFRJ.

Nanda, S. and Chiueh, T. (2015). A Survey on Virtualization Technologies Technical Report. Department of Computer Science, University at Stony Brook, NY.

Públio, A. (2017). Desvendando os modelos de serviços em Cloud (Nuvem): SaaS, PaaS e IaaS https://angelopublio.com.br/modelos-servicos-cloud-saas-paas-iaas, July.

Ribeiro, R. Q.; Foscarini, E.D. (2014). Nuvem UFRGS: IaaS como ferramenta de apoio à pesquisa. In: VIII Workshop de Tecnologia da Informação e Comunicação das IFES, 2014, Brasília.

Rizzetti, B. A. (2015). Virtualização: da teoria a soluções. Trabalho de Conclusão de Curso, Universidade Federal de Santa Maria.

Rocha, L. A. (2013). Introdução à Computação em Nuvem.

Definição de Novas Regras para o IDS Snort em Redes Definidas por Software

Thiago Oliveira Garcia[1], Charles V. Neu[1]

[1]Universidade de Santa Cruz do Sul (UNISC)

thiago.garcia@gaz.com.br,charles1@unisc.br

Abstract. *In this work, we intend to develop new rules for identifying intrusion attacks that can occur in an SDN network through intrusion detection tools that help identify these vulnerabilities, we use the Open Source SNORT system for attack detection, the Mininet emulator to emulate the SDN environment together with the Ryu driver. The results show that the integration between the Ryu controller and Snort was successful, the rules developed in Snort were able to detect attacks against Telnet and FTP, and this same tool sent the alerts to the Ryu controller inside an SDN environment.SDN.*

Resumo. *Neste trabalho buscamos desenvolver novas regras de identificação de ataques de intrusão que podem ocorrer em uma rede SDN através de ferramentas de detecção de intrusão que auxiliem na identificação destas vulnerabilidades, utilizamos o sistema Open Source SNORT para a detecção dos ataques, o emulador Mininet para emular o ambiente SDN juntamente com o controlador Ryu. Os resultados mostram que a integração entre o controlador Ryu e o Snort foi bem sucedida, as regras desenvolvidas no Snort foram capazes de detectar ataques contra Telnet e FTP, e esta mesma ferramenta enviou os alertas ao controlador Ryu dentro de um ambiente SDN.*

1. Introdução

As redes de computadores, tanto tradicionais quanto as Redes Definidas por Software ou *Software Defined Networking* - SDN necessitam de mecanismos de segurança. No modelo tradicional de redes de computadores, a administração dos equipamentos que compõem estas redes é realizada de maneira individual. Este modelo dificulta o trabalho do administrador da rede e torna mais complicada a sua análise para a resolução de um problema ou melhoria na configuração de uma política [Bitencourt 2014]

As redes SDN apresentam uma alternativa para um gerenciamento logicamente centralizado e trazem maior flexibilidade aos administradores de rede, assim facilitando o seu dia a dia. O conceito de redes SDN é proporcionar um novo modelo para administração das redes de computadores, permitindo que um único equipamento possa ter a visibilidade e o controle de toda a rede e uma administração centralizada [Kim 2013]. A utilização deste modelo proporciona também maior flexibilização no controle das redes, onde dividir tipos de tráfego, empregar controles diferentes para cada elemento da rede e analisá-los separadamente é a questão-chave deste novo modelo [Mckeown 2008].

Ao mesmo tempo em que a arquitetura SDN apresenta-se promissora, existem alguns desafios de segurança a serem vencidos nesta nova tecnologia. A centralização do plano de controle traz inúmeras vantagens, como lógica centralizada e visão global da rede [Nadeau 2013]. Tais características representam significativos benefícios, porém aumentam a

exposição do controlador e suas aplicações a ataques na rede e interceptação de fluxos, assim causando um grande transtorno e prejuízo à empresa que foi afetada.

Embora a adoção desta nova tecnologia esteja em fase inicial, através deste trabalho busca-se desenvolver novas regras de identificação de ataques de intrusão que podem ocorrer em uma rede SDN totalmente centralizada, através de ferramentas de detecção de intrusão que auxiliem na identificação destas vulnerabilidades. É utilizado o sistema *Open Source* Snort e o *Dataset* Darpa 99, que contém uma vasta base de ataques com as principais características de cada intrusão. Foi desempenhada uma análise dos dados dos ataques contra os protocolos Telnet e *File Transfer Protocol* - FTP, onde foi determinada uma faixa de tempo de cada ataque e análise dos pacotes de redes baseado em informações que determinam suas características.

O restante deste trabalho está organizado da seguinte maneira: o capítulo 2 apresenta os trabalhos estudados com relação a métodos de segurança em SDN encontradas na literatura. O trabalho desenvolvido é apresentado no capítulo 3. O capítulo 4 mostra os testes realizados e os resultados obtidos. E finalmente, a conclusão e considerações finais são apresentadas no capítulo 5.

2. Trabalhos Relacionados

Mattos *et* al. [Mattos, Ferraz and Duarte 2013] sugere o isolamento da comunicação entre redes virtuais que utilizam uma mesma infraestrutura física que visa garantir a confidencialidade da rede virtual de cada host nesta estrutura física, onde são isolados os recursos de cada host e do tráfego desta rede. Todavia, garantir o isolamento entre estas redes virtuais irá prevenir somente ataques de uma rede virtual em outra rede virtual em uma mesma estrutura física, mas não foca na detecção de ocorrências dos ataques de intrusão dentro do ambiente virtual.

O trabalho desenvolvido por Nagahama [Nagahama 2013], apresenta o IPSFlow, que é uma arquitetura de Sistema de Prevenção de Intrusão que utiliza o SDN e o protocolo OpenFlow para a construção de um sistema de prevenção com ampla cobertura na rede. Nesta arquitetura IPSFlow, de acordo com o resultado da análise feita por um determinado IDS, o controlador OpenFlow teve como bloquear o fluxo de forma automática no switch que está mais próximo da origem do tráfego, impedindo assim, que o tráfego malicioso circule livremente pela rede.

No artigo de Le *et* al. [Le *et* al. 2015], propõem um deslocamento de rede dentro da abordagem de SDN realizando experiências típicas contra os ataques de intrusão, que busca a redução do custo de hardware e software comparado a uma rede tradicional, mantendo o mesmo desempenho. Já no trabalho realizado por Chen *et* al. [Chen and Chen 2015], utiliza um método que pressupõe que o atacante vai usar ferramentas de verificação para fazer um reconhecimento da rede. O autor cria dois algoritmos, o primeiro gera falsos hosts, que espera para o estabelecimento de conexão completa. O segundo algoritmo gera informações falsas, como portas abertas que irão responder ao atacante, sendo uma isca virtual para atrair atacantes em potenciais.

Tabela 1. Comparativo entre trabalhos estudados e trabalho proposto

Trabalho	Snort	Integração	Controlador	Detecção intrusão
Mattos *et* al.	X	x	X	x
Nagahama	✓	x	✓	✓
Le *et* al.	X	x	✓	✓
Chen *et* al.	X	x	X	✓
Este Trabalho	✓	✓	✓	✓

Como pode-se observar na Tabela 1, mesmo aquele que apresenta uma solução entre a ferramenta Snort e o Controlador *Openflow*, o mesmo desenvolveu uma solução própria utilizando os conceitos e características destas mesmas ferramentas. Desta forma, percebe-se que não houve nenhuma tentativa de integrar essas tecnologias sem a necessidade de se criar uma terceira solução para fim de pesquisas.

Este trabalho propõe a integração entre o IDS Snort e o Controlador *Openflow* sem a necessidade de desenvolvimento de uma nova ferramenta. Também é proposta a criação de novas regras de identificação de ataques de intrusão que serão criadas na ferramenta Snort contra ataques de intrusão direcionados aos serviços de Telnet e FTP.

3. Trabalho Desenvolvido

O desenvolvimento deste trabalho foi divido em três etapas. A primeira delas refere-se a forma de como foi realizado a integração entre o IDS Snort e SDN. A segunda etapa foi realizada a coleta e análise dos dados das intrusões destinadas a cada tipo de serviço e por fim, a terceira etapa é a criação detalhada de cada regra conforme alguns métodos escolhidos.

3.1. Integração IDS Snort e SDN

A arquitetura desta integração compreende uma das decisões mais importantes deste trabalho. Diante das limitações apresentadas como a falta de integração entre controladores existentes com dispositivos de segurança, mas especialmente com o IDS Snort para a detecção de ataques de intrusão, foi escolhido então controlador *Ryu.*

Este controlador oferece duas formas para a integração com o Snort, na primeira opção, tanto *Ryu* quanto o IDS trabalham em uma mesma máquina, onde o controlador recebe os pacotes de alertas via *Unix Domain Socket*, que é um mecanismo de comunicação entre processos que permite a troca de dados bidirecional entre processos em execução em uma mesma plataforma. A segunda opção é o controlador e o IDS trabalhar em máquinas diferentes, pois o Snort exige um poder computacional muito alto e o desempenho no ambiente de teste não seria afetado. O *Ryu* recebe os alertas via *Network Socket,* onde é o ponto final da comunicação entre processos e a rede, como por exemplo, a conexão sendo direcionado para uma porta 1234 em um computador localizado no endereço 192.168.2.2 [Ryu 2016].

Entre as duas opções, decidimos como a solução mais adequada sendo a de executar o controlador e o IDS em máquinas diferentes.

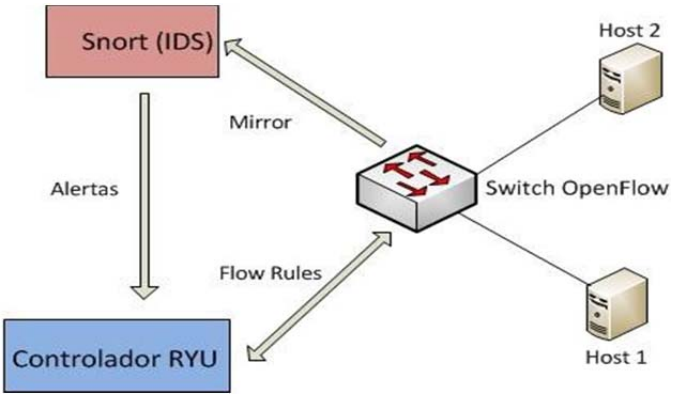


Figura 1. Arquitetura proposta para a Integração entre *Ryu* e Snort

Como ilustrado na Figura 1, para monitorar os pacotes entre os hosts, foi necessário realizar uma configuração no *openflow switch* denominada *Interface Mirror*. Desta forma é possível encaminhar uma cópia de todo tráfego que é transmitido dentro da rede para a interface do Snort, onde o processamento e a verificação das regras acontecerão. Caso alguma regra detecta alguma anormalidade na rede, o IDS enviará alertas ao controlador via *Network Socket*. Depois disso, o *Ryu* poderá tomar decisões na sua rede, assim tendo uma visão mais ampla dos acontecimentos dentro dela. Na figura 1, é ilustrada esta arquitetura.

3.2. Coleta e Análise de Tráfego

Nesta seção, apresentamos a fase da coleta dos dados que foram gerados na rede e a extração das informações que foram necessárias para elaborar as novas regras de intrusão. Para isso, utilizamos duas ferramentas para obter os dados necessários, a ferramenta *THC Hydra* para efetuar ataques contra os protocolos mencionados anteriormente e *Wireshark,* que é uma ferramenta que auxilia na análise do tráfego gerado na rede, e os organiza por protocolos através de uma interface, com a possibilidade de utilizar filtros conforme a necessidade do usuário. A Figura 2 apresenta o ambiente de teste com as ferramentas mencionadas.

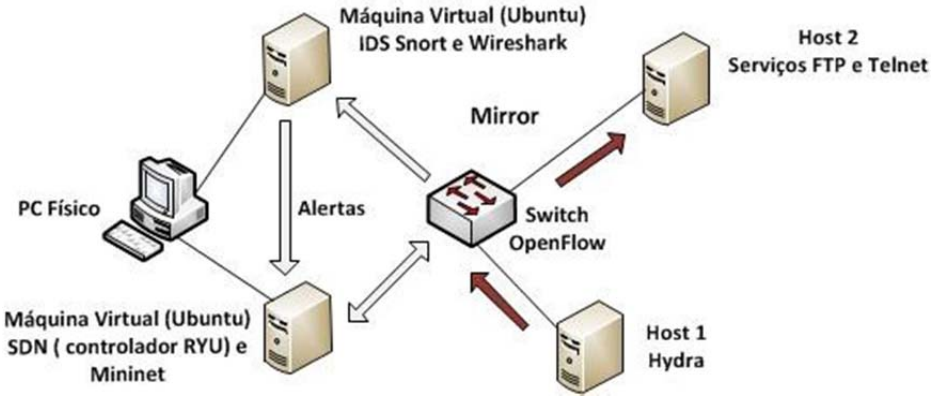


Figura 2. Ambiente de teste

Como mostra na Figura 2, em um primeiro momento, para verificar como um ataque de intrusão se comporta no exato instante em que o mesmo acontece e assim verificar seu funcionamento e seus padrões, utilizou-se a ferramenta *Hydra*, onde se efetuou ataques entre os hosts dentro do emulador Mininet com direção para os protocolos FTP e Telnet.

Para cada tipo de ataque existem alguns padrões, mas para verificar mais afundo cada tipo, foi necessária a utilização do *Wireshark*. Essa ferramenta captura todo o tráfego de uma interface de rede sem ter qualquer interferência no ambiente no qual esta

monitorando. Todo tráfego gerado durante os testes de intrusão realizados pelo *Hydra* e o tráfego normal do ambiente de teste, ficam gravados nesta ferramenta. Utilizando os filtros que a ferramenta fornece, filtramos cada tipo de ataque no qual pretendemos estudar e criamos um repositório para tipo de ataque, utilizando o próprio *Wireshark* para a verificação dos detalhes das intrusões no qual pretendemos criar as regras no Snort.

Tendo estas informações, já é possível a criação de regras no IDS Snort para a prevenção de ataques. Na próxima seção, é explicado como estas informações foram utilizadas na criação das regras dentro do Snort.

3.3. Desenvolvimento de Regras no Snort

Uma das vantagens de integrar Snort com SDN é a flexibilidade na criação das regras que a ferramenta oferece, assim podendo alertar o controlador sobre diversas tentativas de exploração e o mesmo podendo agir conforme sua necessidade. Nesta seção é apresentado o desenvolvimento das regras conforme os métodos propostos, no qual esta dividida em subseção: o desenvolvimento da regra na ocorrência de intrusão direcionada ao protocolo *telnet* esta na subseção 4.3.1. Já na subseção 4.3.2 está o desenvolvimento da regra que verifica a ocorrência de intrusão direcionada ao protocolo FTP.

3.3.1. esenvolvimento Regra na Ocorrência de Tentativa de Intrusão ao Telnet

Para o desenvolvimento da primeira regra, é analisado o tráfego gerado pelo ataque com direção ao protocolo Telnet para descobrir algum padrão que possa ser utilizado para a criação de tal regra dentro do Snort. Foi identificado que o usuário root, no corpo do pacote tanto no valor hexadecimal quanto em ASCII, tem algumas semelhanças que levamos em consideração. Analisando os dados, temos na grande maioria os hexadecimais: 72 6f 6f 74 0d 00. Então utilizamos os seguintes hexadecimais: 72 6f 6f 74 0d 00 no qual foi utilizado dentro da regra.

Também foi verificado o número de *Acknowledgement – ACK*, onde no cabeçalho TCP contém um campo com o número de reconhecimento deste pacote. O campo mostra o próximo número de sequência que o remetente do pacote TCP está esperando para receber. Analisamos qual o valor que mais tem repetição dentro do tráfego gerado pelo ataque, e definimos o valor do ACK em 78.

Após as analises realizadas, definiu-se a seguinte regra:

alert tcp any any -> any 23 (msg:"Possivel ataque Telnet"; flow:to_server; flags: A; ack: 78; content:"|72 6f 6f 74 0d 00|"; rawbytes; offset:0; depth: 6; classtype:bad-unknown; sid:3658011; rev:8;)

Onde **alert tcp any any** é de qualquer origem, **-> any 23** para qualquer destino em direção a porta 23. A definição **(msg:"Possivel ataque Telnet"** é a mensagem no qual o Snort irá mostrar na detecção de alguma intrusão. Para definir o valor de ACK é usado **flow:to_server; flags: A; ack: 78** e que o pacote enviado contenha o hexadecimal **content:"|72 6f 6f 74 0d 00|".** Para que o IDS não verifique todo o pacote é usado o limitador **rawbytes; offset:0; depth: 6**. A regra é classificado como **classtype:bad-unknown** sendo ordenada dentro do snort por **sid:3658011** . **S**ua revisão é indicada como **rev:8;)**.

Então, a regra desenvolvida tem como função identificar qualquer *host* de qualquer rede que tente acessar algum *host* da rede monitorada através da porta 23, em que o ACK seja 78 e o final do cabeçalho contenha os hexadecimais | 72 6f 6f 74 0d 00 |.

3.3.2. esenvolvimento Regra na Ocorrência de Intrusão Direcionada ao FTP

Para o desenvolvimento desta regra, analisamos o tráfego gerado pelo ataque com direção ao protocolo FTP, descobrir o tempo que leva cada tentativa de intrusão e verificar outro padrão que possa ser utilizada para a criação de tal regra dentro do Snort.

Foi verificado que o tempo em que levou cada intrusão é inferior a 1 segundo, independente do usuário no qual esta tentando acesso no *host* alvo. Como a ferramenta utiliza uma base de dados com possíveis chaves de acesso, terá muitas tentativas de conexão em um curto espaço de tempo, menor que 1 segundo.

Foi analisado também o valor do bit PSH flag, onde utilizando o bit 1 ativa esta função, onde informa ao TCP origem que ele deve enviar todos os pacotes, inclusive os que estiverem em sua memoria ao destinatário.

Após as análises realizadas, definiu-se a seguinte regra:

alert tcp any any -> any 21 (msg:"Possivel ataque FTP"; flags: P; threshold: type threshold, track by_dst, count 18, seconds 1; classtype:attempted-recon; sid:3648012; rev:1;)

Onde **alert tcp any any** é de qualquer origem, **-> any 23** para qualquer destino em direção a porta 23. A definição **(msg:"Possivel ataque FTP"** é a mensagem no qual o Snort irá mostrar na detecção de alguma intrusão quando o flag PSH estiver ativo **flags: P;** e se ocorrer 18 tentativas **threshold: type threshold, track by_dst, count 18** dentro de 1 segundo **seconds 1**. A regra é classificado como **classtype:attempted-recon** sendo ordenada dentro do snort por **sid:3648012. S**ua revisão é indicada como **rev:10;)**

Então, a regra desenvolvida tem como função identificar qualquer *host* de qualquer rede que tente acessar algum *host* da rede monitorada através da porta 21, em que a flag PSH esta ativa e que envia mais de 18 requisições no intervalo de 1 segundo.

4. Resultados

Para a validação deste trabalho, foi produzido um ambiente para testar e validar este trabalho, desde a integração entre controlador *Ryu* e o IDS Snort, passando pela análise de tráfego gerado pelas intrusões na rede ate a criação das regras dentro da ferramenta Snort. Estas regras foram desenvolvidas para identificar intrusões dentro de um ambiente de rede controlada, conforme o ambiente de teste apresentado no capítulo anterior.

Para simular os acessos na rede de teste foram utilizados computadores virtuais que receberam endereços IP do range 192.168.41.0/24. Já os hosts criados dentro do emulador Mininet receberam endereço IP do range 192.168.47.0/24. Após, foram realizados diferentes tipos de tentativa de intrusão aos serviços de Telnet e FTP, conforme representado pela Figura 3.

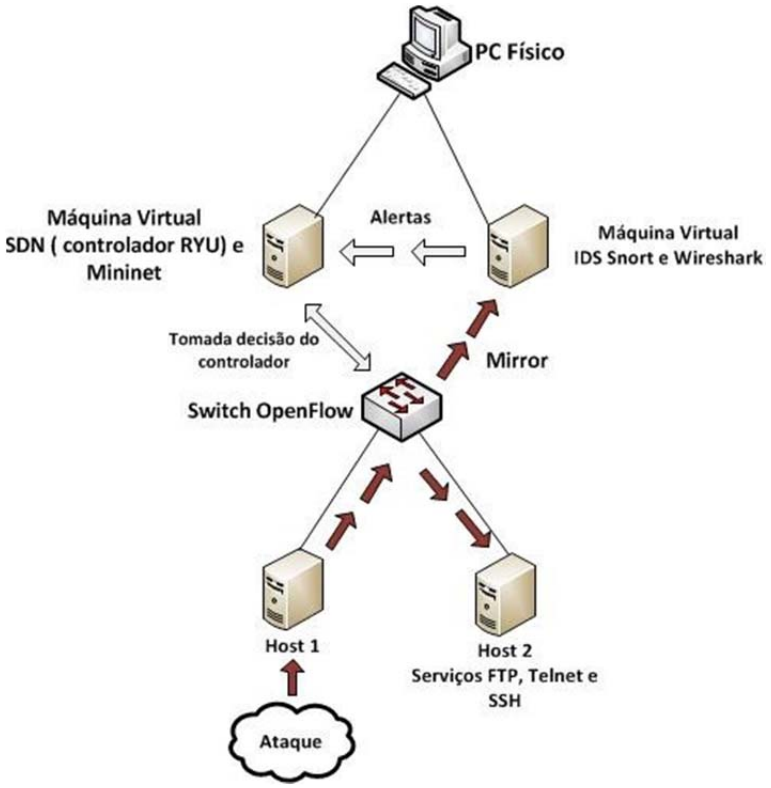


Figura 3. Ambiente desenvolvido para validação e resultados

O ambiente ilustrado pela Figura 3 apresenta o ambiente de teste desenvolvido, onde é identificado o host capaz de realizar algum tipo de intrusão que desrespeite as políticas impostas à rede. Neste cenário o ataque pode ocorrer somente dentro do mesmo ambiente de rede e a atividade é direcionada somente ao host 2.

Na validação das regras de detecção de intrusão, primeiramente utilizamos a ferramenta *Hydra* juntamente com a base de dados que contem diversas senhas para efetuar ataques entre os hosts dentro da rede SDN. Então para efetuar o ataque ao serviço Telnet, utilizamos a porta 23 e para o serviço FTP, a porta 21, no qual é o padrão destes dois serviços. Utilizamos um arquivo no qual continha 3 usuários: root, admin e administrador.

Após, com a ajuda do *Dataset* Darpa 99, inicialmente verificamos os momentos no qual aconteceram os ataques, pois o mesmo disponibiliza um arquivo que detalha todos os acontecimentos desta base, como qual serviço esta sendo atacado e seu tempo de duração. A seguir, com ajuda do *Wireshark* e um editor de texto, separamos os dados em dois diferentes arquivos: rede normal e rede com ataques.

Assim, tendo esses dois cenários, no qual cada um tendo um tráfego diferente foi possível verificar a ocorrência de falsos positivos e falsos negativos em relação às regras criadas no Snort. Desta mesma forma, conseguimos visualizar o Snort tratando o tráfego, analisando estes dados e enviando os alertas ao controlador *Ryu* e assim o mesmo podendo tomar decisões sobre essas requisições dentro do ambiente SDN

Finalizando, é possível afirmar que, dentro do ambiente testado, as regras propostas e seus métodos utilizando Snort, foram capazes de identificar ataques baseado nas regras desenvolvidas e enviar os alertas ao controlador *Ryu* dentro de um ambiente SDN. Da mesma forma, a integração entre as ferramentas propostas cumpriu com seu objetivo, no qual visa o auxilio ao uso futuro de administradores na implantação e gerenciamento dentro do paradigma do SDN.

5. Considerações finais

Através dos resultados obtidos e pelos testes realizados pode-se concluir que as regras criadas através dos métodos pesquisados são capazes de detectar possíveis ataques e assim auxiliar o gestor de rede a protegê-la. Da mesma forma, a integração entre o IDS Snort e o ambiente SDN através do controlador *Ryu* provou ser bem sucedida nos experimentos.

Após monitorar a rede e verificar as regras estabelecidas, o Snort envia os alertas correspondentes para o controlador *Ryu* para que possa tomar a melhor decisão possível em sua rede. A combinação entre estas tecnologias pode revelar-se muito eficaz, uma vez combinadas as vantagens de flexibilidade e controle da rede, podem evitar as vulnerabilidades que esse sistema tem atualmente e proteger contra ataques cibernéticos.

Como melhorias possíveis, utilizando esta integração entre o IDS Snort e o controlador *Ry*u, seria interessante o desenvolvimento de uma aplicação para o efetivo bloqueio destas tentativas de intrusão dentro da rede SDN. Desta forma, mostraria a integração completa entre esses sistemas, desde a geração de alertas através do Snort até o bloqueio deste tráfego malicioso com a utilização do controlador *Ryu* dentro do paradigma SDN.

Outro ponto que pode ser estudado com atenção diz respeito à detecção de outros tipos de intrusão como negação de serviço e a varredura de sistema. Como esta tecnologia ainda existe muitas vulnerabilidades, ela torna-se alvo fácil para um ambiente sem muita proteção, e estes tipos de ataques podem afetar a rede como um todo, pois a negação de serviço tende a deixar um sistema inativo por um período e a varredura de sistema o atacante detectar ainda mais as vulnerabilidades existentes dentro do ambiente de rede.

Referências

Bitencourt, William Lopes (2014). *Implementação de politicas globais em redes SDN utilizando marcação de pacotes*. Unisinos, São Leopoldo.

Chen, P. and Chen, Y. (2015). *Implementation of SDN Based Network Intrusion Detection and Prevention System.* ICCST.

Kim, H. and Feamster, N. (2013) *Improving Network Management with Software Defined Networking.* Communications Magazine, IEEE.

Le, A., Dinh, P., Le, H. and Tran, N. C. (2015). *Flexible Network-based Intrusion Detection and Prevention System on Software-defined Networks*. ICACA.

Mattos, D. M. F., Ferraz, L. H. G. and Duarte, O. B. (2013). *Um mecanismo para isolamento seguro de redes virtuais usando a abordagem hıbrida Xen e OpenFlow*. Em XIII SBSeg, p. 128–141.

Mckeown, N. and Anderson, T. (2008) *OpenFlow:* Enabling Innovation in Campus Networks. ACM SIGCOMM.

Nadeuau, T. D. and Gray, K. (2013). *SDN:* Software Defined Networks. ”O‟Reilly Media, Inc.”, 2013.

Nagahama, Fabio Yu. (2013). *IPSFlow:* um framework para Sistema de Prevenção de Intrusão baseado em Redes Definidas por Software. UFPA.

Ryu (2016). *RYU the Network Operating System.* Disponível em: http://ryu.readthedocs.io/. Acesso em: Setembro de 2016

Arquitetura de comunicação segura para Smart Grids

Yagor S. Duarte[1], Alexandre Silva Rodrigues[2], Bruno da Silva Alves[2], Tiago A. Rizzetti[1], Luciane Neves Canha[2], Marcio de Abreu Antunes[3]

[1]Colégio Técnico Industrial de Santa Maria - CTISM - UFSM
Av. Roraima, 1000 – 97015-900 – Santa Maria – RS – Brasil

[2]Centro de Tecnologia - UFSM – Santa Maria – RS – Brasil

[3]Companhia Estadual de Energia Elétrica - Distribuição - CEEE-D – RS – Brasil

{yagor,alexandre.rodrigues}@redes.ufsm.br, bdalves@inf.ufsm.br,
{tiago.rizzetti, lucianecanha}@ufsm.br, marcioaa@ceee.com.br

Abstract. *The concept of Smart Grids stands out by integrating new technologies and functionalities to the traditional power system. These new technologies bring with them vulnerabilities that need to be processed to obtain a secure channel for information exchange. Thus, it is proposed in this paper, an architecture for secure communication based on the addition of a self-contained device capable of handling the security features in a transparent way, acting as a bridge of communication between the network and the legacy device. In addition, allowing the gradual inclusion of this technology, without affecting the system in operation and at a reduced cost. Tests carried out demonstrate the feasibility of the idea.*

Resumo. *O conceito de Smart Grids se sobressai pela integração de novas tecnologias e funcionalidades ao sistema elétrico tradicional. Essas novas tecnologias trazem consigo vulnerabilidades que necessitam ser tratadas para que exista um canal seguro de troca de informações. Desta forma, é proposta neste trabalho, uma arquitetura para comunicação segura baseada na adição de um dispositivo autocontido capaz de tratar das funcionalidades de segurança de forma transparente, atuando como uma ponte de comunicação entre a rede e o dispositivo legado, assim, permitindo a inclusão gradativa desta tecnologia sem afetar o sistema em funcionamento e a um custo reduzido. Testes realizados demonstram a viabilidade da ideia.*

1. Introdução

O termo Redes Elétricas Inteligentes (REI), é mundialmente debatido como o futuro do Sistema Elétrico de Potência (SEP) atual. Um sistema novo, capaz de introduzir novas tecnologias e funcionalidades aliadas a um melhor gerenciamento de toda a infraestrutura do sistema de energia. Para obter o efeito esperado, faz-se necessária a implementação de um mecanismo de comunicação confiável.

A incorporação das tecnologias da informação e redes de comunicação bidirecionais também no segmento de distribuição do SEP irá proporcionar um conjunto de novas aplicações, entre elas a infraestrutura de medição e atuação remota denominada *Advanced Metering Infrastructure* (AMI). Muitas mudanças são esperadas com a

implementação da AMI. Dentre elas vale destacar, como exemplo, o gerenciamento de demanda. Com essa nova capacidade de acompanhar e controlar a demanda por energia, possivelmente a curva de carga dos consumidores será mais homogênea, por meio de uma tarifação dinâmica [Brown 2008] [Yan et al. 2013]. Com essas novas possibilidades de medições mais precisas, será possível estabelecer tarifações para diferentes horários do dia, a partir dos níveis de consumo. Isso significa que, em períodos do dia que cotidianamente apresentem um elevado consumo, poderá ser adotada uma tarifa de maior custo. [Rivera et al. 2013] [Yan et al. 2012].

Devido a racionalização da geração e demanda de energia, incluindo novas possibilidades de aplicações, surge a necessidade de evoluções tecnológicas com o objetivo de obter um melhor gerenciamento e evitar falhas. Esse novo sistema deve apresentar um eficiente mecanismo de comunicação digital. Os dispositivos devem poder responder a comandos remotos para recuperar ou melhor distribuir energia elétrica aos consumidores [CGEE 2012].

Falhas de sistema não devem representar grandes impactos aos consumidores, concessionárias e ao sistema elétrico de potência como um todo. Portanto, é imprescindível que o sistema apresente um certo nível de resiliência, sendo capaz de se manter em operação, mesmo na presença de falhas controladas e até mesmo retornar a seu estado estável de operação. Essas particularidades devem estar inclusas nas tecnologias de informação e mecanismos de comunicação de dados presentes neste ambiente. Todo o controle de operações e reconfigurações sobre o SEP é baseado nos dados provindos do sensoriamento, como o dos dispositivos de medição inteligente, que devem trocar informações em um ambiente seguro.

Os recursos computacionais de hardware e software essenciais para a implementação do conceito de REI, estarão incorporados de forma embarcada nos novos dispositivos a serem projetados. Porém, faz-se necessária uma gradativa implementação deste conceito ao SEP, ou seja, adicionar essas tecnologias aos dispositivos legados já presentes no sistema atual. Isso justifica-se devido ao elevado tempo que seria necessário para a substituição imediata de todos os equipamentos do SEP, além do, economicamente inviável, investimento financeiro que teria de ser feito para tais mudanças.

Pensando nisso, este artigo traz uma proposta de uma arquitetura de comunicação segura onde um dispositivo do tipo caixa preta será utilizado na comunicação entre o sistema de supervisionamento e os dispositivos para adicionar algumas funcionalidades aos novos e expandir as capacidades dos legados. Dentre os recursos de software disponibilizados, estará a utilização de plugins para tratamento dos pacotes, como proteções criptográficas, antes de chegarem na rede de comunicação.

Nas próximas seções será apresentada a arquitetura proposta neste artigo, suas motivações e resultados obtidos.

2. Arquitetura proposta

2.1. Sistema SCADA

Um sistema SCADA pode ser visto como um sistema de automação ou controle industrial, que através de protocolos de comunicação pode monitorar, controlar e se comunicar com sensores e atuadores. Desta forma, efetuando leituras de informações, ou

mesmo enviando comandos para estes dispositivos[Strehl 2012]. Em função da diversidade de dispositivos presentes nesse tipo de sistema, os protocolos utilizados são variados. A rede de comunicação utilizada também é heterogênea, abrangendo sistemas legados como o MODDBUS, bem como sistemas mais atuais que utilizam redes baseadas em IP [Ghansah 2009] [Makhija and Subramanyan 2003]. Devido à extrema relevância das informações em uma *Smart Grid*, são essenciais capacidades de segurança em tempo real ao sistema SCADA, como integridade e autenticidade dos pacotes [Aloul et al. 2012] [Ghansah 2009].

O sistema SCADA obtém os dados através de uma rede IP dos seus dispositivos supervisionados [Knapp and Langill 2014]. A adoção de padrões de protocolos abertos, baseados no protocolo IP, expande o contexto de segurança. Possíveis ataques ao sistema SCADA, como ataques de negação de serviço, mascaramento de ip, espionagem e falsificação de dados, seriam de alto impacto ao SEP. Ataques podem fazer com que os dados tornem-se inválidos ou inacessíveis, dessa forma, as aplicações que fazem uso desses dados podem deixar de operar, causando danos ao sistema de energia. Com esse cenário, podem ser verificadas as vulnerabilidades no sistema de comunicação.

Alguns sniffers de rede, como o Wireshark ou TCPDump podem ser utilizados para realizar a análise das informações de tráfego de rede. Informações críticas, como as oriundas dos medidores inteligentes, podem ser interceptadas e adulteradas caso não exista algum tipo de criptografia na mensagem. Além do fato de que ataques de negação de serviço ao sistema de supervisionamento podem torná-lo indisponíveis.

2.2. Arquitetura desenvolvida

Com base nessas vulnerabilidades de segurança, soluções foram desenvolvidas com o objetivo de atenuá-las, utilizando como referência os prazos de tempo de mensagem, estabelecidos pelo padrão IEC61850 [Mackiewicz 2006]. A adição de assinatura digital e criptografia, que exigem um certo poder de processamento, será provida por meio do dispositivo tipo caixa preta. Esse dispositivo contem suporte, via hardware e software, para essas funcionalidades.

Padrões como o IEC 61850 e IEC 62351 apresentam, respectivamente, arquiteturas de mensagens e segurança, visando padronização na implementação de Smart Grids [Baigent et al. 2004] [Cleveland 2012]. Porém, a implementação de segurança não é bem especificada. Existem incompatibilidades de hardwares, que possuem limitada capacidade computacional, mensagens com prazos diferentes, e muitos IEDs implementam ainda protocolos legados como o MODBUS.

Nesse contexto, para prover funcionalidades de segurança aos dispositivos, faz-se necessária a utilização de componentes adicionais na arquitetura. Neste artigo, optou-se pelo uso do Mediador de Pacotes (MEPA) e o uso de plugins de tratamento de pacotes. Esses visam implementar requisitos que supram as normas do padrão de segurança [Hohlbaum et al. 2010]. Com a utilização desses, a comunicação entre o sistema SCADA e os IEDs serão realizadas de forma segura. Passando a solucionar as questões referentes à confidencialidade, integridade, disponibilidade e autenticidade.

2.2.1. Plugins

Para prover as funcionalidades de tratamento dos pacotes, serão utilizados plug-ins de acordo com a necessidade de operação. Um plugin é um elemento de software carregado a partir do software principal, neste caso o MEPA. O plugin é responsável por realizar o tratamento sobre o pacote de rede, não demandando qualquer modificação ou recompilação do código da aplicação principal. Desta forma, cada nova demanda no tratamento de pacotes poderá ser adicionada através da implementação de um plugin apropriado.

Uma vez que o pacote seja tratado, ele é devolvido à aplicação principal que irá deixar que este siga o fluxo normal. Desta forma, para que o tratamento do pacote possa ser efetivo ele deverá ser realizado em ambas as partes comunicantes, ou seja, emissor e receptor. Os elementos envolvidos no caminho da comunicação não sofrem qualquer alteração.

No caso da segurança da comunicação, por exemplo, deverá ser desenvolvido um plugin responsável por implementar um sistema de chaves criptográficas para garantir autenticidade, confidencialidade e integridade dos dados. Junto ao emissor e receptor uma instância deste plugin deve ser executada, permitindo a comunicação entre as partes. Na arquitetura proposta os plugins são carregados pelo MEPA e executam sobre a plataforma de hardware adicionada para tal, no formato de um dispositivo do tipo caixa preta, autocontido e transparente aos elementos finais nos quais se conecta.

2.2.2. MEPA

Mediador de Pacotes ou MEPA, foi construído a partir da utilização do driver TUN/TAP, disponível no Kernel do Linux [Maxim Krasnyansky]. Ele é responsável por realizar a filtragem de pacotes, verificando se está presente no pacote informações que correspondam a algum dos filtros previamente cadastrados. O software MEPA deve ser implementado no dispositivo autocontido, representado junto ao IED e ao sistema SCADA. Caso o MEPA não encontre nenhum filtro que corresponda ao pacote interceptado, ele irá reinserir o pacote no mesmo ponto da pilha TCP/IP do kernel. Essa característica torna a arquitetura flexível, uma vez que somente serão afetados tráfegos específicos de uma aplicação, representada pelo filtro cadastrado.

As funcionalidades a serem providas pelo MEPA são basicamente 3: a) Interceptar pacotes na interface de entrada; b) Verificar se o pacote corresponde a algum filtro pré-cadastro, em caso afirmativo passar ao plugin correspondente e; c) reinserir o pacote na pilha de protocolos TCP/IP para que seja devidamente tratado pela pilha de protocolos disponível no kernel Linux. Nota-se que a reinserção de pacote deverá ser realizada tanto se o pacote foi tratado por um plugin como quando ele não corresponde a nenhum filtro e, portanto, nenhum tratamento é realizado. A figura 1 ilustra esse processo.

As características do MEPA tornam a proposta flexível para poder ser utilizada em qualquer cenário de comunicação entre sistemas SCADA e IEDs. Em dispositivos de softwares que podem ser modificados, o MEPA e os plugins podem ser implementados de maneira embarcada.

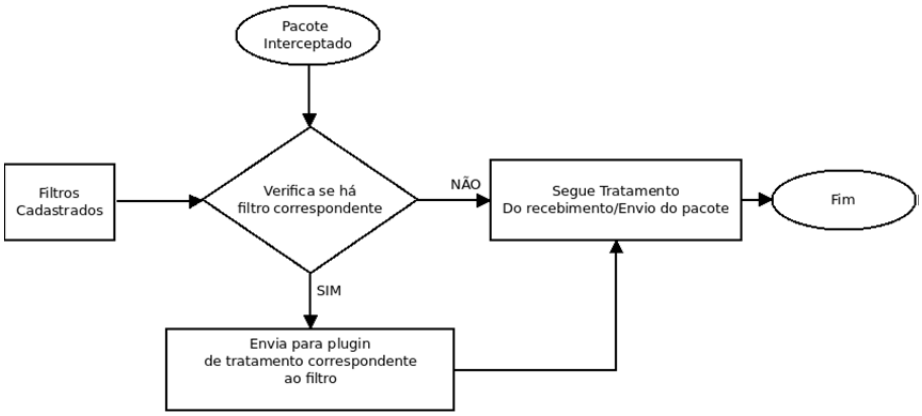


Figure 1. Cenário da arquitetura proposta (Fonte: Acervo pessoal).

A função de interceptação de pacotes do MEPA é proporcionada pela utilização do plugin TUN/TAP, disponível no Linux. TUN/TAP é um driver presente no Kernel do Linux, capaz de criar interfaces de rede virtuais. Tais interfaces permitem que ao invés de receber/enviar pacotes através de um hardware, o dispositivo receba/envie pacotes provenientes de uma aplicação em modo usuário (*user space*).

O driver TUN/TAP provê dois tipos de dispositivos: a) TAP: interface capaz de operar na camada de enlace, enviando/recebendo pacotes da Ethernet; b) TUN: interface capaz de operar na camada de rede, enviando/recebendo pacotes do tipo IP (*Internet Protocol*) [Maxim Krasnyansky]. Assim, através do driver TUN/TAP, uma aplicação é capaz de enviar ou receber pacotes da pilha de protocolos de rede, permitindo que a aplicação presente no MEPA seja capaz de interceptar, repassar, modificar e analisar os pacotes passantes.

3. Resultados parciais obtidos

Para verificar a eficiência e viabilidade da proposta apresentada nesse trabalho, criou-se um ambiente de testes, utilizando-se um cenário real, composto por um sistema SCADA e um painel fotovoltaico. Esse painel possui possibilita o monitoramento e gerenciamento remoto, obtidos através da utilização do sistema SCADA. Nesses termos, para realizar o monitoramento foi utilizado o software *Elipse Power*, que possibilita monitorar as informações referentes à energia gerada pela placa em tempo real.

A troca de mensagens entre a placa fotovoltaica e o sistema SCADA foi realizada por meio do protocolo MODBUS sobre TCP, ou seja, após estabelecer uma comunicação inicial em uma porta específica da placa, o sistema SCADA realiza a sondagem de informações a cada intervalo de tempo preestabelecido. Os dados dessa leitura são apresentados em uma interface HMI (*Human Machine Interface* ? Interface Homem-Máquina), na qual o técnico pode intervir no funcionamento caso esses dados estejam fora do padrão. Também há possibilidade de que o sistema SCADA possa intervir automaticamente, quando programado para tal função, diante de políticas de funcionamento.

Quanto aos aspectos práticos relacionados ao cenário de teste, foram utilizados dois computadores, os quais executavam o MEPA (para interceptar e analisar os pacotes) e um plugin desenvolvido para realizar operações de criptografia e descriptografia de mensagens que estavam de acordo com os padrões definidos nos filtros utilizados no MEPA.

Dos computadores utilizados o primeiro identificado como PC_i5 é constituído de um processador i5 da Intel® de 3320M de cache com *clock* de 2.6GHz, 8 GiB de memória DDR3 de 1600 MHz. O segundo computador identificado como PC_AMD, é constituído de um processador AMD Phenom II X2 B55 de 7256M de cache com *clock* de 1,5GHz, 4 GiB de memória DDR3 de 1333MHz.

Com base no cenário de testes descrito anteriormente, foi realizado um ataque de Homem do Meio, inserindo pacotes modificados, pacotes falsos e aplicando retardos na comunicação. Este ataque consiste na inserção de um novo elemento na comunicação de preferência entre os dispositivos que estão com a comunicação estabelecida e de modo transparente. A medida que os pacotes eram inseridos na comunicação, o MEPA interceptava-os e redirecionava para o plugin verificar os aspectos relacionados à criptografia e assinatura digital. Dessa forma, os pacotes modificados e pacotes falsos foram descartados, já que suas características estavam alteradas, o plugin não conseguiu descriptografar, nem validar sua assinatura digital.

Outro aspecto importante que deve ser abordado para garantir a segurança e disponibilidade de um sistema é a proteção contra ataques de negação de serviço (DoS, *Denial of Service*). Sendo assim, para analisar o comportamento da arquitetura proposta, utilizou-se a ferramenta HPING3 para simular um ataque DoS no sistema SCADA e painel fotovoltaico, efetivando uma inundação de requisição SYN na porta especifica do sistema SCADA. Nesse sentido, como em toda e qualquer comunicação, antes de chegar ao sistema SCADA, essas requisições foram interceptadas pelo MEPA e redirecionadas para a verificação de segurança. Dessa forma, a partir da resposta negativa do plugin, o MEPA descartou todas requisições indevidas. Além disso, todas as mensagens com criptografia incorreta, quebra de integridade do pacote e/ou confidencialidade foram descartadas.

Além disso, optou-se por verificar o tempo necessário para realizar operações de criptografia e descriptografia dos pacotes interceptados pelo MEPA. Os resultados obtidos referem-se ao tempo que cada computador precisou para realizar tais tarefas. Para realizar esse teste, foram realizadas sessões de cem sondagens ao sistema SCADA à placa fotovoltaica, efetuadas 10 vezes em diferentes horários do dia, totalizando 1000 sondagens. A média de tempo que o computador PC_i5 levou em cada sondagem, foi de 4,09 milissegundos para criptografar e descriptografar os dados. A média de tempo que o computador PC_AMD levou foi de 10,97 milissegundos.

Assim sendo, com a utilização do plugin integrado ao MEPA nas duas extremidades realizando a verificação de segurança, os pacotes demoraram em média 15,06 milissegundos a mais em uma sondagem. A Figura 2 apresenta o gráfico que mostra a notável diferença de desempenho de um computador para outro. O computador PC_i5 levou menos tempo para aplicar a segurança à informação do que o computador PC_AMD. Isso acontece porque os computadores utilizados possuem características diferentes, impactando no tempo em que cada um necessita para realizar as tarefas. Pode-se afirmar que no contexto de Redes Elétricas Inteligentes, isso é um problema comum, pois há heterogeneidade de dispositivos com limitações de processamento e memória.

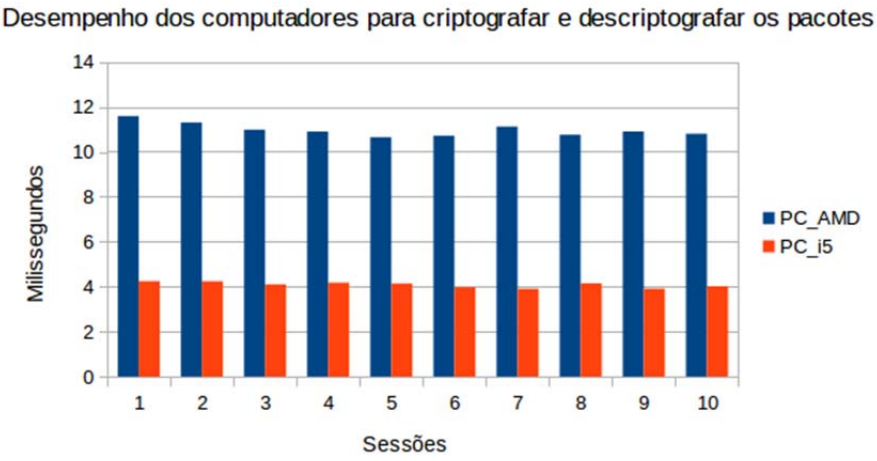


Figure 2. Gráfico de desempenho dos computadores utilizados nos testes (Fonte: Acervo pessoal).

4. Conclusão

A implementação do conceito de REI em sistemas de energia é uma realidade, assim como, as preocupações com a segurança da comunicação desses sistemas. Neste trabalho, as principais tecnologias utilizadas pelo sistema, suas ameaças e vulnerabilidades foram levantadas. Diante disso, para amenizar parte desses problemas, propôs-se a utilização de um Mediador de Pacotes, que possibilita a integração com plugins capazes de realizar operações voltadas a garantir a segurança das informações trocadas entre equipamentos presentes no sistema elétrico e os sistemas supervisórios. Nesses termos, a arquitetura apresentada possibilita implementar tais medidas em qualquer cenário que utilize o sistema SCADA. Com isso, todos os equipamentos dessa rede podem utilizar este serviço.

Desta forma, a comunicação entre o sistema SCADA e IEDs passa a ser assinada digitalmente ou criptografada, provendo integridade, legitimidade e autenticidade das mensagens. Além disso o Mediador de Pacotes controla todas as comunicações de entrada e saída do sistema SCADA e dos IEDs. Nesse contexto, a utilização do Mediador de Pacotes e do plugin desenvolvido demonstraram ser uma solução viável, apresentando resultados promissores, ou seja, por meio da arquitetura proposta é possível garantir que os dados cheguem autênticos e íntegros.

Durante os testes no cenário real, observou-se um discreto retardo nos tempos necessários para aplicar criptografia ou assinatura digital na comunicação. No entanto, pode-se afirmar que diante da diversidade de dispositivos utilizados em Redes Elétricas Inteligentes, é possível implementar segurança em mensagens cuja comunicação acontece através de uma comunicação TCP/IP cliente/servidor e também atender os prazos de tempo, especificados no padrão IEC 61850 em mensagens MMS. Para trabalhos futuros sugere-se a otimização do código do Mediador de Pacotes, para melhorar a performance e modificá-lo para que realize a aplicação de assinatura digital e ou criptografia em mensagens com prazos de tempo reduzido especificadas pelo padrão IEC 61850.

Agradecimentos

Os autores agradecem o apoio da ANEEL P&D Código PD-5707-4301/2015, CEEE-D, UFSM e CNPq (Processo 311516/2014-9).

Referências

Aloul, F., Al-Ali, A., Al-Dalky, R., Al-Mardini, M., and El-Hajj, W. (2012). Smart grid security: Threats, vulnerabilities and solutions. *International Journal of Smart Grid and Clean Energy*, 1(1):1–6.

Baigent, D., Adamiak, M., Mackiewicz, R., and SISCO, G. M. G. M. (2004). Iec 61850 communication networks and systems in substations: An overview for users. *SISCO Systems*.

Brown, R. E. (2008). Impact of Smart Grid on Distribution System Design.

CGEE (2012). Centro de gestão e estudos estratégicos (org.). *Redes elétricas inteligentes: contexto nacional. Brasília: Tatiana de Carvalho Pires*.

Cleveland, F. (2012). Iec tc57 wg15: Iec 62351 security standards for the power system information infrastructure. *White Paper*.

Ghansah, I. (2009). Smart grid cyber security potential threats, vulnerabilities and risks. *California Energy Commission, PIER Energy-Related Environmental Research Program, CEC-500-2012-047*.

Hohlbaum, F., Braendle, M., and Alvarez, F. (2010). Cyber security practical considerations for implementing iec 62351. In *Proceedings of the PAC World Conference*.

Knapp, E. D. and Langill, J. T. (2014). *Industrial Network Security: Securing critical infrastructure networks for smart grid, SCADA, and other Industrial Control Systems*. Syngress.

Mackiewicz, R. (2006). Overview of iec 61850 and benefits. In *Power Systems Conference and Exposition, 2006. PSCE'06. 2006 IEEE PES*, pages 623–630. IEEE.

Makhija, J. and Subramanyan, L. (2003). Comparison of protocols used in remote monitoring: Dnp 3.0, iec 870-5-101 & modbus. *Electronics Systems Group, IIT Bombay, India, Tech. Rep*.

Maxim Krasnyansky, Maksim Yevmenkin, F. T. Universal TUN/TAP device driver. `https://www.kernel.org/doc/Documentation/networking/tuntap.txt`. Accessed: 2017-07-20.

Rivera, R., Esposito, A. S., and Teixeira, I. (2013). Redes elétricas inteligentes (smart grid): oportunidade para adensamento produtivo e tecnológico local. *Revista do BNDES 40*, pages 43–84.

Strehl, L. C. (2012). Prospecção de tecnologias para aumentar a segurança em sistemas scada.

Yan, Y., Qian, Y., Sharif, H., and Tipper, D. (2012). A Survey on Smart Grid Communication Infrastructures : Motivations , Requirements and Challenges. 15(1):1–16.

Yan, Y., Qian, Y., Sharif, H., and Tipper, D. (2013). A survey on smart grid communication infrastructures: Motivations, requirements and challenges. *IEEE Communications Surveys Tutorials*, 15(1):5–20.

Estudo comparativo entre depuradores para SDN

Thales Nicolai Tavares[1], Anderson Monteiro da Rocha[1], Leonardo da Cruz Marcuzzo[1], Nilton Camargo Batista da Silva[1], Vinicius Fülber Garcia[1], Carlos Raniery Paula dos Santos[1]

[1]Departamento de Computação Aplicada PGCC – Universidade Federal de Santa Maria (UFSM)
Av. Roraima nº 1000 – 97.105-900 – Santa Maria – RS – Brasil

{tntavares,amonteiro,lmarcuzzo,nbatista,vfulber,csantos}@inf.ufsm.br

Abstract. *An implementation of Software Defined Networks allows the benefit of network technology professionals as an alternative to centralized network management. In addition, SDN emits a central programmable controller, enabling new and developed network applications. Instead of a control schedule, it also presents a possibility to introduce errors in the applications, as well as an unwanted behavior. To mitigate problem, there are tools called debuggers, as banks perform testicles for errors. This work proposes to perform a comparative study among SDNs debuggers, where they are presented as features, advantages and disadvantages.*

Resumo. *A implementação de Software Defined Networks permite benefícios aos profissionais da área de tecnologia de redes, como a alternativa de gerenciamento centralizado da rede. Além disso, SDN apresentam um controlador central programável, possibilitando que novas aplicações de rede sejam desenvolvidas. Ao proporcionar a programabilidade do controlador, também apresenta a possibilidade de introduzem erros nas aplicações, assim ocasionando um comportamento não desejado. Para mitigar problema, existem ferramentas chamadas depuradores, as quais realizam testes em busca de erros. Este trabalho tem o proposito de realizar um estudo comparativo entre depuradores SDNs, onde serão apresentadas as características, vantagens e desvantagens.*

1. Introdução

Com o crescimento da internet e sua utilização de forma comercial, a expansão das redes se deu de forma inevitável, porém a tecnologia de arquitetura baseada no modelo TCP/IP não acompanhou o crescimento no que se refere a sua inovação, além disso, o fato dos dispositivos serem controlados por softwares proprietários, engessou qualquer forma de desenvolvimento e personalização para a configuração das redes.

Consequentemente, surge as Redes Definidas por Software (SDN), que apresenta um novo conceito para a estrutura das redes que traz mais flexibilidade, rapidez e favorece novos serviços [Nadeau and Gray 2013]. Sendo assim, as Redes Definidas por Software representam uma nova maneira de olhar a forma como as redes são controladas, configuradas e operadas.

As SDN permitem maior flexibilidade, pois são controladas por softwares [Riggio et al. 2013]. Ao invés vez dos consoles de gerenciamento de redes e comandos

que exigem um grande esforço operacional, tornando complexa a administração em larga escala. Nesse novo conceito, pode-se começar a depurar redes assim como um software. Sendo assim, temos com objetiva apresentar um estudo comparativo entre depuradores para redes definidas por software e expor suas características.

O presente trabalho esta organizado de forma que a Seção 2 apresenta uma revisão sobre os temas envolvidos no trabalho, a Seção 3 apresenta os depuradores utilizados. A Seção 4 apresenta o ambiente os cenários envolvidos nos testes, na Seção 5 são apresentados os resultados e as considerações finais sobre o trabalho é apresentado na Seção 6.

2. Redes Definidas por Software

A rede definida pelo software é a maneira de abordar a rede de computadores através de abstrações de *software* em lugar de hardware especializado [Nadeau and Gray 2013]. Ao abstrair algumas das funcionalidades de baixo nível da rede em aplicações, permite que os administradores de rede gerenciar com mais facilmente as redes dinâmicas.

Nesse contexto, as rede definida por software separa a parte da infraestrutura de rede que determina onde a informação está sendo enviada, plano de controle, da parte onde os dados realmente se movem, plano de dados [Baldini et al. 2012]. Assim e permite a parte de tomada de decisão acontecer na aplicação.

2.1. OpenFlow

Afim de permitir realizar experimentos de novas propostas para redes de computadores em escalas menores, foi proposto o Comutador *OpenFlow* [Heller 2009], o qual possui a função de possibilitar seja executado experimentos em uma rede real. Nesse sentido a plataforma *OpenFlow* tem como objetivo criar um ambiente de rede de teste programável, unindo as qualidades da virtualização de redes com o conceito de Redes Definidas por Software [Guedes et al. 2012].

O protocolo *OpenFlow* apresenta grande vantagem em sua utilização, pois é possível realizar testar em infraestruturadas sem que interfira no tráfego real da rede de produção [McKeown et al. 2008], sem a necessidade que os fabricantes de comutadores exponham os projetos de software e hardware dos seus equipamentos. *OpenFlow* emprega o conceito de redes definidas por software, cujo o substrato físico é composto pela parte que cuida do tráfego de produção da rede e outra parte que cuida do tráfego experimental das novas propostas de rede. Assim, a plataforma *OpenFlow* procura oferecer uma opção controlável e programável, porém não pode-se descartar a possibilidade conter trechos de códigos com problemas [Canini et al. 2012].

3. Depuradores de rede

As redes de computadores possuem um elevado nível de dificuldade de depurar [Handigol et al. 2012]. Atualmente os gerentes de redes dispões de um conjunto de ferramentas como *tcpdump* para monitoramento dos dispositivos finais e *netflow* em *switches* e roteadores [Reitblatt et al. 2011]. Realizar a tarefa de depurar um rede é considerada árdua, pois as ferramentas buscam reconstruir o estado da rede de maneira ad-hoc, no entanto protocolos de camada 2 e 3 possuem mudanças de estados contínuos [Handigol et al. 2012]. Em Redes Definidas por Software ferramentas que possibilitam

realizar depuração de uma rede, da mesma maneira que realiza-se depuração em software [Handigol et al. 2012].

A ferramenta *OFRewind* é um depurador que possui o objetivo de auxiliar os gerentes de rede a localizar erros presentes em uma redes com *OpenFlow*, além de possibilita realizar a tarefa de análise da rede [Heller et al. 2013]. Ja a ferramenta NICE faz-se eficaz ao depurar programas OpenFlow, pois representa a verificação de um modelo e uma execução simbólica para encontrar erros em programas *OpenFlow* [Canini et al. 2012].

4. Caso de teste

- **OFRewind:** O cenário utilizado para realizar o teste para avaliação do OFRewind, será o *Anomalous Forwarding*, sendo baseado na documentação desenvolvidos pelos desenvolvedores. Para a elaboração dos testes, foram utilizados alguns *switches*, cada um com dois computadores conectados, conforme ilustrado na Figura 1 apresentada na Sessão 5.
- **NICE:** O cenário utilizado para realizar o teste para avaliação do NICE, será o *Pyswitch*, que é uma aplicação *OpenFlow* que tem como objetivo implementar o registro de endereços MAC Address, junto com o processo de *flooding* para destinos desconhecidos, muito comum em *Switches Ethernet*.

5. Resultados e Discussões

Com o proposito de atender os objetivos propostos neste trabalho, explorando os depuradores presentes para SDN, assim como suas características, faz-se necessário, para o melhor compreensão, estipular parâmetros de comparação entre essas ferramentas. Os parâmetros definidos são: instalação, usabilidade, documentação disponível, número de casos de teste disponíveis e eficiência.

5.1. OFRewind

Conforme os critérios definidos, os resultados do estudo e teste desta ferramenta são:

- **Instalação:** Não se aplica.
- **Usabilidade:** Embora seja uma ferramenta *Open source*, existem limitações. Por não conseguir retorno com o desenvolvedor, não foi possível obter o código fonte para compilar a ferramenta.
- **Quantidade de caso de teste:** Em sua documentação apresenta consigo três diferentes casos de teste, o qual possibilita o conhecimento das de suas funcionalidades.
- **Documentação disponível:** A ferramenta dispõe de uma documentação satisfatória, que com obtenção do código fonte, proporciona uma melhor elaboração de casos de teste para serem realizados nesta ferramenta.
- **Violações Encontradas:** Conforme documentação disponível, torna-se possível obter informações de casos de teste já realizado e documentados.

Para averiguar o desempenho de dispositivos com base de um novo fornecedor (fornecedor C), foi registrado o tráfego de um fluxo específico. Para afim de teste foi enviado um *ping* com 10 segundos de atraso entre um par de *hosts* conectados ao *switch* fornecedor

B. Logo foi usado o recurso de mapeamento de dispositivos e portas do *Ofreplay* para reproduzir o tráfego do host c7 para o c8 pela porta 8 e 42 pertencente ao *switch* fornecedor C na Figura 1.

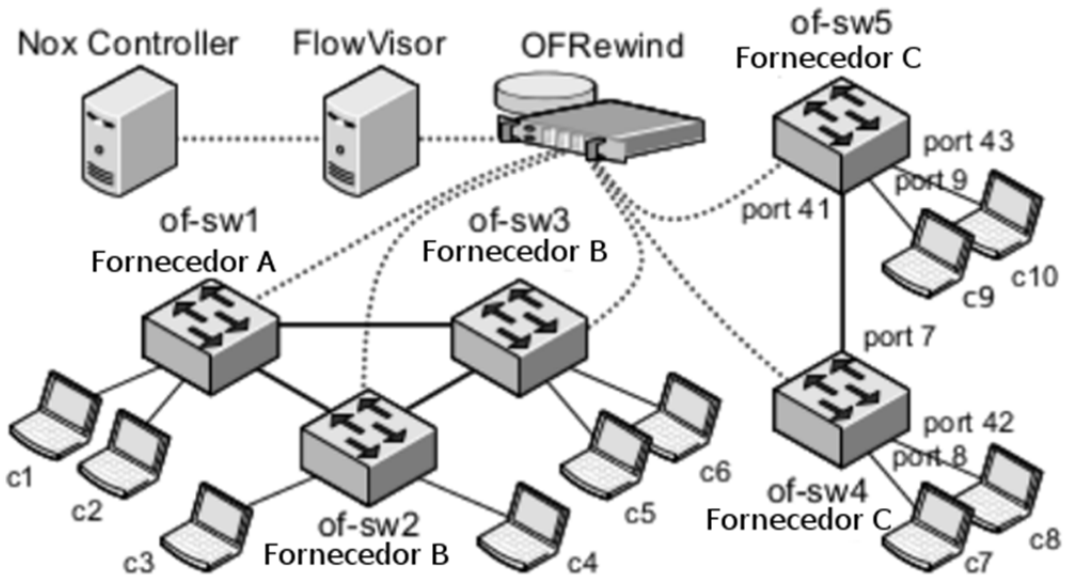


Figure 1. Topologia para testes do OFRewind

Logo depois da repetição, podemos percener uma limitação no switch fornecedor C. O fluxo do ping no perioldo de resolução do ARP, os mesmos transmitidos do host c7 são recebidos pelo host c10, mas não pelos host c8 c9. Na tabela 1 é apresentada a entrada da de fluxos criado no switch of-sw4. Conclui-se que a ação de Flooding não está devidamente aplicada pelo switch fornecedor C.

CONTADOR	AÇÃO
duration=181s	in_port=8
n_packets=0	dl_type=arp
n_bytes=3968	dl_src=00:a1:f9:32:b6:44
idle_timeout=60	dl_dst=ff:ff:ff:ff:ff:ff
hard_timeout=0	actions=FLOOD

Table 1. Tabela de entrada de fluxos *switch* of-sw4

5.2. NICE

Conforme os critérios definidos, os resultados do estudo e teste desta ferramenta são:

- **Instalação:** O processo de instalação é encontrado na página[1] dos desenvolvedores.Se faz essencial alguns *plugins* adicionais, como o Python com versão 2.6 ou superior.

[1]https://code.google.com/p/nice-of/wiki/DocRequirements

- **Usabilidade:** Embora o NICE não oferecer uma interface gráfica, a sua usabilidade é satisfatória. Por meio de apenas poucos comandos é possível realizar o teste da aplicação desejada. Como exemplo, para realizar o caso de teste *pyswitch*, é somente executar o NICE através da chamada de comando: ”*./nice.py config/pyswitch.conf* ”.
- **Quantidade de caso de teste:** Esta ferramenta disponibiliza somente três casos de testes, mas não impossibilita que seja adicionado novos testes pelo usuário.
- **Documentação disponível:** É disponibilizado em sua página[2] materiais sobre a ferramenta, o qual simplifica a pesquisas e o desenvolvimento, além de um manual completo da utilização deste depurador. Também é disponibilizados três artigos[3][4][5], em fóruns internacionais, sobre esta ferramenta.
- **Violações Encontradas:** Com a realização do teste padrão do NICE (pyswitch) e o resultado aprensentado na Figura 2, faz-se capaz de confirma a ocorrencia de tres falhas relatadas na documentação do NICE.

```
--- Results ---
Total states: 3102
Unique states: 1520
Revisited states: 1582
Maximum path length: 34
Invariant violations: 35
NoLoop              : 0 violations
internal check      : 0 violations
NoDrop              : 2 violations (first found after 13.17s, 1114 transitions)
StrictDirectRoute   : 28 violations (first found after 13.55s, 1155 transitions)
NoForgottenPackets  : 5 violations (first found after 13.17s, 1114 transitions)
ReturnContinueStop  : 0 violations
:: Symbolic engine quitting...
```

Figure 2. Resultados execução pyswitch.conf

5.3. Comparações

As comparações foram realizadas com base na utilização das ferramentas e com base na documentação disponibilizada Afim de que se tenha uma melhor compreensão, é apresentado em tabelas as comparações. Sendo a Tabela 2 apresenta os controladores suportados por cada depurador. Já as características analisadas dos depuradores obtidas através das pesquisas aqui realizadas, é apresentada na Tabela 3.

DEPURADOR	POX	NOX	TESTE DE SWITCH
OFRewind	não	sim	sim
NICE	sim	sim	não

Table 2. Compatibilidade depuradores

[2]https://code.google.com/archive/p/nice-of/
[3]http://infoscience.epfl.ch/record/170618
[4]http://infoscience.epfl.ch/record/169211
[5]http://infoscience.epfl.ch/record/167777

	EFICIÊNCIA	INSTALAÇÃO	USABILIDADE	DOCUMENTAÇÃO
OFRewind	bom	ruim	não se aplica	ótima
NICE	ótimo	bom	ótimo	ótimo

Table 3. Análise dos depuradores

6. Considerações Finais

Com base da pesquisa efetuada, foi possível constatar que Redes Definidas por Software apresenta uma ampla aceitação em meios acadêmicos e corporativos. O desenvolvimento do protocolo *OpenFlow* teve imensa relevância para a expansão das Redes de computadores. A proposta ofertada pelo protocolo *OpenFlow* concede diversos avanços em pesquisas abrangendo redes de computadores,a capacidade de segmentar o plano de dados e o plano de controle, é uma grande vantagem do uso de Redes Definidas por Software. A partir deste mecanismo se faz capaz de realizar teste de ambientes e configurações de redes sem a necessidade de que a rede fique fora de operação ou apresente irregularidade em seu funcionamento.

References

Baldini, G., Sturman, T., Biswas, A. R., Leschhorn, R., Godor, G., and Street, M. (2012). Security aspects in software defined radio and cognitive radio networks: A survey and a way ahead. *IEEE Communications Surveys & Tutorials*, 14(2):355–379.

Canini, M., Venzano, D., Peresini, P., Kostic, D., and Rexford, J. (2012). A nice way to test openflow applications. In *Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, number EPFL-CONF-170618.

Guedes, D., Vieira, L., Vieira, M., Rodrigues, H., and Nunes, R. V. (2012). Redes definidas por software: uma abordagem sistêmica para o desenvolvimento de pesquisas em redes de computadores. *Minicursos do Simpósio Brasileiro de Redes de Computadores-SBRC 2012*, 30(4):160–210.

Handigol, N., Heller, B., Jeyakumar, V., Maziéres, D., and McKeown, N. (2012). Where is the debugger for my software-defined network? In *Proceedings of the first workshop on Hot topics in software defined networks*, pages 55–60. ACM.

Heller, B. (2009). Openflow switch specification.

Heller, B., Scott, C., McKeown, N., Shenker, S., Wundsam, A., Zeng, H., Whitlock, S., Jeyakumar, V., Handigol, N., McCauley, J., et al. (2013). Leveraging sdn layering to systematically troubleshoot networks. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 37–42. ACM.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J. (2008). Openflow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74.

Nadeau, T. D. and Gray, K. (2013). *SDN: Software Defined Networks: An Authoritative Review of Network Programmability Technologies*. " O'Reilly Media, Inc.".

Reitblatt, M., Foster, N., Rexford, J., and Walker, D. (2011). Consistent updates for software-defined networks: Change you can believe in! In *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, page 7. ACM.

Riggio, R., Rasheed, T., and Granelli, F. (2013). Empower: A testbed for network function virtualization research and experimentation. In *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*, pages 1–5. IEEE.

Análise de Desempenho e de Dados de um Provedor de Internet

Marcos Z. de Mello, Cristiano Bertolini[1], Edison Pignaton de Freitas[2], Evandro Preuss[1], Ricardo Tombesi Macedo[1]

[1]Universidade Federal de Santa Maria - UFSM
Campus Frederico Westphalen - Frederico Westphalen - RS

[2]Universidade Federal do Rio Grande do Sul - UFRGS
Caixa Postal 15064 Porto Alegre - RS

zanonmarcos@mail.ufsm.br, cristiano.bertolini@ufsm.br,

epfreitas@inf.ufrgs.br, evandro.preuss@gmail.com, rmacedo@inf.ufsm.br

Abstract. *The Internet is present in our daily life and it is essential for communication and entertainment. However, Internet providers in the countryside of Brazil have a poor and limited connection in comparison with big cities. This paper presents an analysis of user profiles from an Internet provider of Brazil countryside and the traffic generated by them in order to know the coherent plans with the consumption of data of the clients. Based on this analysis it was proposed a recommendation system and then the Internet provider can offer the most suitable plan.*

Resumo. *A Internet está cada vez mais presente no nosso dia a dia e se torna essencial para comunicação e entretenimento. No entanto, provedores de Internet no Brasil fornecem uma internet muitas vezes limitada em termos de velocidade, principalmente em locais distantes dos grandes centros urbanos. Outro problema é uma possível limitação do acesso por meio de uma franquia de dados. Este artigo apresenta uma metodologia de análise de perfis de clientes de um provedor de Internet no interior do Brasil e os tráfegos gerados por estes de modo a saber se os provedores irão oferecer planos coerentes com o consumo de dados dos clientes. Com base nesta análise foi proposto um sistema de recomendação para que a empresa possa fornecer o pacote de dados mais adequado aos seus clientes.*

Palavras-Chave: Provedores de Internet, Franquia de Dados, Tráfego de Dados.

1. Introdução

Muito se discute sobre a inserção de novos pacotes de Internet Banda Larga Fixa com franquia de dados. Um dos problemas dessa implantação consiste em uma falta de análise do tráfego gerado pelos clientes mensalmente, definindo perfis de acordo com o consumo e a velocidade contratada, para averiguar se os pacotes oferecidos são coerentes com seus clientes.

Nos últimos anos, a tecnologia *streaming* que é uma forma de transmissão instantânea de dados de áudio e vídeo se desenvolveu no Brasil pela melhora na velocidade das conexões com a Internet. Por meio de serviços de *streaming on-demand* é possível assistir filmes ou escutar músicas sem a necessidade de fazer download. Isso possibilita que o usuário esteja no controle do que vai assistir e quando irá assistir. Tais serviços já são tarifados em rede de Internet móvel por meio de uma franquia de dados, que é o volume de dados que pode ser trafegado na rede no pacote contratado, que inclui as informações enviadas e recebidas.

Este artigo apresenta uma metodologia de estudo do perfil de uso de internet aplicada a clientes de um provedor de internet Tchê Turbo interior do Rio Grande do Sul e propõe um sistema de recomendação para que os gestores do provedor de Internet possam adaptar melhor os seus planos ao seus clientes. A metodologia se baseia no uso de métricas que possibilitam analisar o tráfego de dados gerado na rede do provedor de acesso. Estas métricas referem-se a uma avaliação do uso de dados e banda dos aplicativos mais utilizados, segundo dados da pesquisa realizada em dezembro de 2015 do CONECTA [1]. A análise dos clientes foi realizada durante um ano e definiu o perfil dos clientes ativos por meio do tráfego gerado na rede, mapeando o uso de seus dados e sua respectiva velocidade contratada em um intervalo de tempo. O diagnóstico permitiu que fosse desenvolvido um sistema de recomendação, a fim de se tornar o principal aliado ao sistema de vendas de novos pacotes de internet fixa com franquia de dados, maximizando as opções de comercialização para a empresa Tchê Turbo Provedor de Internet.

As principais contribuições desse trabalho são: (i) um estudo de clientes ativos de um provedor de Internet analisando consumo de dados e planos contratados; (ii) proposta de uma metodologia de coleta e análise de dados de perfis de uso de clientes de provedores de Internet; e (iii) uma proposta de um sistema de recomendação para análise de tráfego e franquia de dados.

2. Estudo de Caso: Provedor de Internet Tchê Turbo

A largura de banda *(Bandwidth)* de uma rede é a medida da capacidade de transmissão em um certo período de tempo, determinando a velocidade que os dados passam através desta rede específica. A largura de banda depende estritamente do meio de transmissão. Esta medida é calculada através do número de bits transportada em um dado intervalo de tempo. Por exemplo, se a rede tem uma largura de banda de 5 milhões de bits/segundo (Mbps), significa que ela é capaz de entregar 5 milhões de bits a cada segundo, ou seja, quanto maior a quantidade de dados trafegando maior terá que ser a capacidade de banda [Peterson and Davie 2013]. Em consequência disso, fala-se, nos dias atuais muito sobre a Computação em Nuvem *(cloud computing)*, que possibilita o acesso de arquivos, além de executar diferentes tarefas de qualquer computador ou dispositivo móvel que tenha acesso à Internet, sendo que os dados não se encontram em uma máquina específica, mas sim em uma rede. Dessa forma surge a tecnologia *Streaming* que torna as conexões mais rápidas, pois possibilita o envio de informações multimídia, através de transferência de dados [Taurion 2009].

A empresa Provedor de Internet Tchê Turbo, disponibiliza o acesso comercial para os clientes por meio de contratos de Internet Banda Larga, no contexto de uma possível

[1]http://www.conectaibrasil.com.br/noticias/pesquisa-do-conectai-revela-quais-sao-os-apps

inclusão de pacotes de dados nos planos. Por muitas vezes um indivíduo não possui conhecimentos técnicos necessários para realizar escolhas entre as várias opções que lhe são apresentadas. Diante deste problema, se faz necessário confiar nas recomendações que são sugeridas por outras pessoas que tenham mais experiência profissional. Nesse mesmo âmbito no início da década de 90, foi desenvolvido o primeiro sistema de recomendação, em seguida uma base de informações que proporcionou maior credibilidade do que a recomendação humana [Pimentel and Fuks 2012].

2.1. Análise dos clientes

A Figura 1, apresenta a análise dos clientes da empresa Tchê Turbo refente aos planos mais comercializado no período de 1 ano, sendo eles os planos de: 1Mb, 2Mb, 3Mb, 5Mb e 8Mb, o qual Mb significa *Megabit*. Por meio desta análise foi possível determinar a quantidade do tráfego de dados gerado na rede pelos usuários obtendo um média e uma mediana da soma total de cada largura de banda.

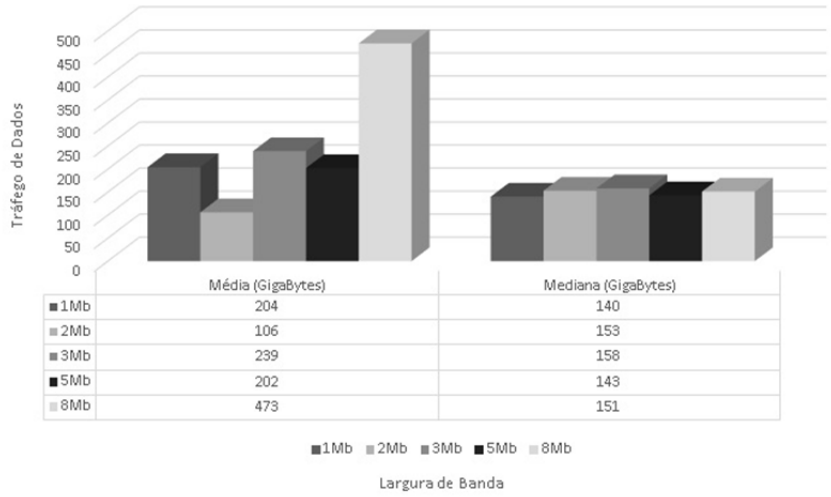


Fig. 1. Analise mensal de tráfego de dados.

Os dados apresentados na Figura 1 diferenciados por linhas no gráfico, colunas e cores identificam respectivamente: As linhas em horizontal demonstram a quantidade de dados gerados em *GigaByte*; Os tamanhos das colunas apontam os níveis de diferença entre um plano e outro do tráfego de dados em *GigaByte*. Por fim as cores equivalem na tabela assim como no gráfico as Larguras de Bandas avaliadas no período de um mês.

Adotou-se a média para que houvesse um único valor que representa-se o trafego que os clientes geram por mês referente a sua largura de banda. Entretanto como houve uma pequena porcentagem de clientes que geram mais tráfego e outros menos, usou-se a mediana para ter um valor mais preciso em relação à média. Avaliou-se no período de Janeiro à Dezembro do mesmo ano, sendo que:

- 420 clientes com largura de banda contratada de 1 *Megabit*. Obteve-se uma média de trafego anual de 2453 GB e uma mediana de 1689 *GigaByte*. De acordo com os dados.
- 1049 clientes com largura de banda contratada de 2 *Megabit*. Obteve-se uma média de trafego anual de 1279 *GigaByte* e uma mediana de 1840 *GigaByte*.

- 159 clientes com largura de banda contratada de 3 *Megabit*. Obteve-se uma média de trafego anual de 2866 *GigaByte* e uma mediana de 1904 *GigaByte*.
- 282 clientes com largura de banda contratada de 5 *Megabit*. Obteve-se uma média de trafego anual de 2422 *GigaByte* e uma mediana de 1719 *GigaByte*.
- 508 clientes com largura de banda contratada de 8 *Megabit*. Obteve-se uma média de trafego anual de 5674 *GigaByte* e uma mediana de 1820 *GigaByte*.

2.2. Análise da Conectividade de Aplicativos

Foram analisados os aplicativos mais utilizados pelos Brasileiros no ano de 2015, a fim de obter um valor de tráfego de dados que as aplicações geram e sua respectiva velocidade, conforme mostra a Tabela 1. Para este estudo usou-se dois aplicativos de monitoramento de rede, o *GlassWire* e o *My Data Manager*. O *GlassWire* é um monitor de rede que visualiza a atividade de rede atual por tipo de tráfego e largura de banda, e gera gráficos com os indicativos. Por sua vez o *My Data Manager* é um aplicativo para dispositivos móveis que apenas gera gráficos de consumo de dados, ou seja, somente o que foi trafegado na rede em determinado horário.

Tabela 1. Avaliação dos aplicativos.

Aplicativos	Trafego de Dados	Largura de Banda
Netflix	885,6	6
YouTube	377,1	4
Skype	301,6	2
Mapas	282,6	0,4
Facebook	111,3	2
Spotify	86,2	4
Twitter	65,3	2
Linkedin	60,7	3

A Tabela 1 apresenta os resultados da avaliação do tráfego de dados e largura de banda através do aplicativo de monitoramento *GlassWire* dos aplicativos em uma hora de uso contínuo. O estudo consiste e uma medida por meio do monitoramento da quantidade de tráfego que gera cada aplicativo utilizado, assim como sua respectiva largura de banda que se faz necessária para o funcionamento da mesma. Para confirmar a integridade do aplicativo de monitoramento o primeiro teste foi realizado com o *Netflix* que fornece serviços de *streaming* de vídeo, que com uma resolução média gerou 885,6 *MegaBytes* de tráfego de dados e necessitou de uma velocidade de 6 *MegaBits* assim como informa o próprio site do *Netflix*.

A Tabela 2 apresenta os resultados da avaliação do tráfego de dados através do aplicativo de monitoramento *My Data Manager* dos aplicativos em uma hora de uso contínuo.

3. Sistema de Recomendação: Metodologia Proposta

A metodologia que é composta por etapas: **Etapa 1**: consiste em realizar uma análise de tráfego de dados e largura de banda dos aplicativos mais usados. Cada aplicativo é avaliado de forma individual por um período de tempo de uma hora. **Etapa 2**: consiste em avaliar a média do tráfego de dados gerados pelos clientes. Para isso é analisado os clientes

Tabela 2. Avaliação do tráfego dos aplicativos.

Aplicativos	Trafego de Dados
Istagram	135
Snapchat	132
WhatsApp	39
Loja de aplicativos	30,6
Waze	9,9
Jogos	6,2
Bancos	4,1

de cada plano comercializados na empresa, referentes a largura de banda e posteriormente realizado uma média anual e uma média mensal do tráfego gerado. **Etapa 3**: consiste em recomendar um plano com franquia de dados para os clientes. Esta recomendação é realizada por meio de um cálculo que depende da escolha dos aplicativos e do tempo de uso pelo cliente.

Foram propostas duas fórmulas para o sistema de recomendação baseadas nas análises preliminares e representadas por equações. A Equação 1 resulta em um valor para a recomendação de planos. Os valores resultantes ficam entre 100 e 1000000 *MegaBytes*. A equação é definida como:

$$Resultados = \left(\sum_{i=m}^{n} Aplicativo_i \right) * QuantidadeHoras * 30 \quad (1)$$

onde *Resultados* representa o resultado da operação, *SomatorioApps* é a soma dos valores em *MegaBytes* levantados na avaliação dos aplicativos e *QuantidadeHoras* é a quantidade de que o usuário fica conectado por dia, multiplicado por 30 dias. O resultado gerado por meio da equação 1 deverá estar de acordo com análise feita do perfil dos clientes, se *Resultados* for igual a 280000 *MegaByte* recomendará um plano, caso *Resultados* for igual a 0, então as opções não foram selecionadas.

A Equação 2 apresenta a soma de todos os aplicativos avaliados neste estudo:

$$\sum_{i=m}^{n} Aplicativo_i := Aplicativo_m + Aplicativo_{m+1} + \ldots + Aplicativo_n \quad (2)$$

onde $\sum_{i=m}^{n} Aplicativo_i$ considera o somatório dos dados analisados de n aplicativos utilizados pelo cliente.

4. Sistema de Recomendação: Implementação

A recomendação foi desenvolvida por meio de métricas que usam os dados de tráfego de dados assim como a largura de banda. A sugestão do plano para o cliente se dá por intermédio de perguntas ao mesmo, no qual tem que responder quais aplicativo usa no dia a dia e por quanto tempo, que quando selecionados pelo vendedor retorna a recomendação.

O Algoritmo 1 apresenta o pseudo código para recomendação de planos com intuíto de auxilar a comercialização dos mesmos. A linha 1 é definida pelo início do al-

gorítimo; a linha 2 define o acumulador *multiplica* iniciando com o valor 1; a linha 3 define o acumulador *somadosvalores* iniciando com o valor 0; a linha 4 refere-se a uma condição que se recebido os valores da seleção dos aplicativos então executará as instruções; a linha 5 verifica se existe valores dos aplicativos, se houver valores, executa a instrução; a linha 6 realiza a soma quando houver valores de mais que um aplicativo e guarda em um acumulador; a linha 7 é uma condição que se não houver valores, executa a instrução a seguir; a linha 8 informa que não existe nenhum valor; a linha 9 refere-se a uma condição que se recebido os valores da seleção dos aplicativos então executará as instruções; a linha 10 verifica se existe valores dos aplicativos, se houver valores, executa a instrução; a linha 11 realiza a multiplicação quando houver o valor de uso dos aplicativos diariamente e multiplica pela quantidade de dias no mês e posteriormente guarda em um acumulador; a linha 12 é uma condição que se o valor da instrução da linha 11 for maior que 100 MB e menor ou igual a 80000 MB a 13 sugere um plano; A linha 14 é uma condição que se o valor não se enquadrou na primeira condição, irá percorrer até encontrar a condição adequada.

Algorithm 1 Algoritmo de Recomendação

```
1:  procedure SisRecRegras
2:      multiplica = 1
3:      somadosvalores = 0
4:      if recebe valores then
5:          if valores = app then
6:              somadosvalores = (somadosvalores + value)
7:          else
8:              print "Nao existe nada selecionado"
9:      if recebe valores then
10:         if valores = multiplica then
11:             multiplica = (multiplica ∗ value) ∗ 30
12:     if multiplica > 100 e ≤ 80000 then
13:         print "Recomenda-se:  3Mb de Velocidade e 80GB de trafego de dados mensais"
14:     else
15:     if multiplica > 80000 e ≤ 210000 then
16:         print "Recomenda-se:  8Mb de Velocidade e 210GB de trafego de dados mensais"
17:     else
18:     if multiplica > 210000 e ≤ 380000 then
19:         print "Recomenda-se:  16Mb de Velocidade e 380GB de trafego de dados mensais"
20:     else
21:     if multiplica > 380000 e ≤ 440000 then
22:         print "Recomenda-se:  24Mb de Velocidade e 400GB de trafego de dados mensais"
23:     else
24:     if multiplica > 440000 e ≤ 1000000 then
25:         print "Recomenda-se:  32Mb de Velocidade e 680GB de trafego de dados mensais"
```

Com base no Algorítimo 1 desenvolveu-se um sistema web que é possível selecionar as opções de uso dos aplicativos informadas pelos clientes, com base nessas escolhas foram realizadas duas simulações. A primeira simulação foi de um perfil de cliente que usa moderadamente a Internet, apontando os aplicativos que foram selecionados, assim como o tempo de navegação, de modo que o botão enviar, serve para que as informações selecionadas sejam mescladas e calculadas. Este primeiro exemplo, os aplicativos usados são *WhatsApp*, *Facebook* e *Youtube*, com um tempo estimado de usabilidade de 2 horas, com estes aplicativos. Bem como apresenta a recomendação do plano, baseado nas escolhas feitas pelos usuários, onde cada aplicativo tem um valor de trafego por hora, sendo assim realizou-se o cálculo que WhatsApp tem tráfego de 39 *MegaBytes*, *Facebook* de 111 *MegaBytes* e *YouTube* de 377 *MegaBytes* por hora. A soma desses valores multipli-

cado por 2 que seria o tempo de navegação selecionado e por 30 que é quantidade de dias no mês. A largura de banda para a recomendação é baseada na análise feita dos aplicativos. O resultado obtido nesse cálculo recomendou o plano de 3 Mb e 80 GB de tráfego de dados mensais.

Em um segundo exemplo de utilidade de aplicativos de um perfil de cliente que usa de forma normal a Internet. Pode ser observada no caso o WhatsApp, *FaceBook*, *Instagram*, *Netflix*, *Snapchat*, *Spotify* e *Linkedin* com um tempo estimado de usabilidade de 5 horas, com estes aplicativos. Bem como apresenta a recomendação do plano, baseado nas escolhas feitas pelos usuários, onde cada aplicativo tem um valor de tráfego por hora, sendo assim realizou-se o cálculo que *WhatsApp* tem tráfego de 39 *MegaBytes*, *Facebook* de 111 *MegaBytes*, *Instagram* de 135 *MegaByte*, *Netflix* 885 *MegaBytes*, *Snapchat* de 132 *MegaBytes*, *Spotify* de 86 *MegaByte* e *Linkedin* 60 *MegaBytes* por hora. A soma desses valores multiplicado por 5 que seria o tempo de navegação selecionado e por 30 que é quantidade de dias no mês. A largura de banda para a recomendação é baseada na análise feita dos aplicativos. O resultado obtido por esse cálculo recomendou o plano de 16 Mb e 32 GB de tráfego de dados mensais.

5. Trabalhos relacionados

[Wilma E. Rosado 2016] propôs recomendações de características ergonômicas para Interfaces de Sistemas de Monitoramento de Redes Baseadas em critérios de usabilidade. O problema relacionado às interfaces de monitoramento de redes atuais, são técnicas e não apresentam facilidade de uso para usuários com pouco conhecimento técnico. Desse modo a solução transpõe que um conjunto de recomendações de características ergonômicas baseadas em critérios de facilidade de utilização de sistemas de monitoramento, através de uma avaliação de interfaces de monitoramento utilizadas pela sociedade de gerência e operação de redes aplicada a uma metodologia que utiliza ferramentas de medidas quantitativas, que divididas em três etapas, a de levantamento de perfil a fim de identificar os usuários, atribuição de pesos a cada recomendação por grau de importância e avaliação de conformidade de cada recomendação, de modo que utilizando as informações anteriores para calcula-se o grau de adequação. O resultado deste trabalho sucedeu-se na adequação, de forma que seja feita a recomendação de um plano mais coerente aos clientes.

[Bivas Bhattacharya 2015] apresenta uma proposta de um *Software* com controlador embutido nos dispositivos móveis (MD) para implementar aos usuários uma política que fornece preferências, diante de escolhas nas aplicações. Porém quando vários usuários ou aplicativos compartilham a mesma rede a largura de banda precisa ser compartilhada entre eles. Nisso o TCP é aplicado pois garante a equidade entre os diferentes usuários ou aplicativos. A solução para casos onde um canal sem fio se degrada, é a tecnologia de rede *Software Defined Network* (SDN), propondo incorporar um *switch* virtual e um Controlador de SDN dentro do MD. Assim as saídas seriam implicadas e compartilhada em vários canais virtuais possibilitando garantia de uma banda miníma.

Observou-se nesses trabalhos a necessidade do controle de tráfego analisado em dois grupos: Cliente e Servidor. Do ponto de vista do cliente os trabalhos apresentaram avaliações de tráfego em determinados horários de uso, observando qual é o comportamento do cliente mediante seu plano contratado. Por outro lado, na perspectiva do servi-

dor, os trabalhos analisam a quantidade de tráfego gerado pelos usurários, com o intuito de verificar se os provedores estão preparados para uma grande quantidade de tráfego simultaneamente. Por fim apresentou uma metodologia que avalia o tráfego dos clientes por largura de banda, avaliando quanto de trafego gera, assim como a largura de banda que utiliza com o uso dos aplicativos indicados pelos clientes e com base nessas avaliações demonstra um cálculo para a recomendação de plano mais coerente ao perfil do cliente.

6. Conclusões

Neste artigo apresentou uma proposta de metodologia de recomendação de novos pacotes de planos com franquia de dados para de provedores de Internet baseado no estudo de caso da empresa Tchê Turbo Provedor de Internet. A metodologia proposta é constituída por três etapas, sendo a primeira de Análise dos aplicativos, a segunda de Análise dos perfis dos clientes e por fim um cálculo que gera a recomendação dos planos.

Com a adequação à proposta de recomendação de planos, a empresa terá um ganho significativo: desde a sua abordagem ao cliente, além de recomendar algo mais coerente com o perfil do usuário. A empresa poderá analisar novamente os dados sempre que necessário para criar e adequar planos de largura de banda prevendo a inserção de pacotes de franquia de dados. Por meio da análise dos clientes, observou-se que a grande maioria dos avaliados possuem planos com largura de banda inferior ao recomendado. Assim, tornando-se um ponto a ser trabalhado com o setor comercial do provedor de Internet, pois se hoje a franquia de dados entrar em vigor 11,27% dos clientes avaliados sentirão as consequências devido ao tráfego gerado mensalmente com base na Largura de banda contratada. O sistema de recomendação proposto possui como principais vantagens: a recomendação aos clientes por velocidade, e isso também auxilia na indicação de planos aos novos Clientes. Outra vantagem foi identificar os clientes com um consumo excessivo de dados, e que poderão em no futuro sofrerem com a limitação da franquia de dados.

Um dos trabalhos futuros é o desenvolvimento de um sistema de predição de consumo de dados pelos clientes. Tal sistema poderia ser desenvolvido com base no estudo já realizado e utilizando-se técnicas de Inteligência Artificial para prever o crescimento do consumo de dados pela empresa.

Referências Bibliográficas

Bivas Bhattacharya, D. D. (2015). Software Defined Network Controller Embedded in Mobile Device for User's Policy Implementation. In *International Conference on Industrial Instrumentation and Control (ICIC)*. IEEE.

Peterson, L. L. and Davie, B. S. (2013). *Redes de computadores: uma abordagem de sistemas*. Elsevier Editora Ltda, 5 edition.

Pimentel, M. and Fuks, H. (2012). *Sistemas Colaborativos*. Elsevier Editora Ltda.

Taurion, C. (2009). *Computação em Nuvem: Transformando o Mundo da Tecnologia da Informação*. Brasport.

Wilma E. Rosado, Leobino N. Sampaio, J. A. S. M. (2016). Recomendações de Características Ergonômicas para Interfaces de Sistemas de Monitoramento de Redes Baseadas em Critérios de Usabilidade. In *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*.

Avaliação Experimental dos *Brokers Apache Flume* e *Kafka* no Contexto de *Big Data*

Matheus Orlandi de Castro[1], Cristiano Bertolini[1], Edison Pignaton de Freitas[2], Evandro Preuss[1], Ricardo Tombesi Macedo[1]

[1]Universidade Federal de Santa Maria - UFSM
Campus Frederico Westphalen - Frederico Westphalen - RS

[2]Universidade Federal do Rio Grande do Sul - UFRGS
Caixa Postal 15064 Porto Alegre - RS

mo.castro@hotmail.com, cristiano.bertolini@ufsm.br, epfreitas@inf.ufrgs.br, evandro.preuss@gmail.com, rmacedo@inf.ufsm.br

Abstract. *Recently, many enterprises are using the Big Data technologies as a new market niche. However, there is no consensus about that is the most appropriated broker to employ in the Big Data context. In the literature, the main approaches are evaluating proprietary solutions, focusing only on processing, or they are targeting other contexts. This paper presents a experimental performance evaluation of the open sources Apache Flume and Kafka brokers. The evaluation considers the data extraction and data transfer processes on the big data context. The results reveal Apache Flume overcoming Kafka on 19% working with small files and on 10% handling big files. However, Kafka overcomes Apache Flume by handling real time collected streaming data.*

Resumo. *Atualmente, muitas empresas vêm explorando tecnologias de Big Data como um novo nicho de mercado. No entanto, não existe um consenso sobre qual broker é mais indicado parar utilizar no contexto de big data. Na literatura, os principais trabalhos avaliam soluções proprietárias, se preocupam apenas com questões de processamento ou se concentram em contextos diferentes. Este trabalho apresenta uma avaliação de desempenho experimental do processo de extração e transferência de dados dos brokers de código aberto Apache Flume e Kafka no contexto de big data. Os resultados obtidos revelam que o broker Apache Flume supera seu concorrente em 19% ao trabalhar com arquivos pequenos e em 10% ao manipular arquivos grandes. Entretanto, o broker Kafka supera em 21,6% seu concorrente ao manipular dados de streaming coletados em tempo real.*

1. Introdução

Big Data surgiu como um nicho de mercado para muitas empresas que prestam serviços através da Internet [Pääkkönen and Pakkala 2015]. *Big Data* consiste em uma área de pesquisa para abordar o grande crescimento do volume de dados, desde a sua geração, passando pela aquisição, análise e armazenamento. De acordo com Gantz e Reinsel, o volume de dados criados e copiados aumentou nove vezes nos

cinco anos anteriores, representando um total de 1,8 ZB até o ano de 2011 (Zettabyte) [Gantz and Reinsel 2011]. Estudos recentes apontam que este volume dobrará a cada dois anos até 2020 [Gantz and Reinsel 2012]. Por meio do *big data* torna-se possível analisar grandes quantidades de dados, extraindo informações que podem trazer lucros para uma determinada corporação ou uma melhora significativa na gestão de recursos e o desenvolvimento de novas soluções explorando este conceito. Algumas das empresas mais conhecidas que exploram esta tendência consistem no *Twitter*, *LinkedIn* e *Facebook*.

No entanto, um desafio atual consiste na escolha de um *broker* de dados mais apropriado para implantação soluções baseadas em *big data*. Uma arquitetura *big data* geralmente compreende três fases principais, a fase de extração, transferência e processamento de dados [Pääkkönen and Pakkala 2015]. Um *broker* consiste em um componente de software responsável por coletar, transportar e negociar dados de uma base de dados origem para um mais clientes [Ramachandran et al. 2005]. Existem muitas soluções de *broker* disponíveis, no entanto, não existe um consenso sobre qual solução consiste na mais indicada para utilizar no contexto de *big data*.

Na literatura, os principais trabalhos avaliam o desempenho de soluções proprietárias, focam na fase de processamento de dados dos *brokers* ou se concentram em contextos diferentes do *big data*. Henjes *et. al.* avaliaram o desempenho da vazão do *Message Broker WebSphereMQ* [Henjes et al. 2006]. Todavia, apenas uma solução foi avaliada, a qual possui o código fechado. Córdova avaliou duas das plataformas de código aberto na área de processamento de *big data*, *Storm* e *Spark* [Córdova 2014]. Todavia, o estudo foi conduzido para analisar apenas a fase de processamento de dados dos *broker*, ignorando as etapas de extração e transferência de dados. Ionescu avaliou os *brokers* de código aberto *RabbitMQ* e *ActiveMQ* em situações de aumento do tamanho da mensagem [Ionescu 2015]. Todavia, o escopo da pesquisa foi restrito ao contexto de tecnologias de *middleware* orientados a mensagens, não compreendendo o contexto de *big data*.

Este trabalho apresenta uma avaliação de desempenho experimental dos *brokers Apache Flume* e *Kafka* no contexto de *big data*. Para conduzir a avaliação foi desenvolvido um protótipo de um *software* capaz de simular um fluxo grande de dados e também realizar trocas de mensagens entre *brokers*. Diferentemente dos trabalhos da literatura, este trabalho avalia a fase de extração e transferência de dados. Além disto, este trabalho considera apenas soluções de código aberto que podem ser livremente adaptadas para fins específicos e são amplamente utilizados em grandes provedores de serviços, como o *LinkedIn*, *Amazon*, *Netflix*.

Os resultados obtidos mostram que a escolha de qual destes softwares depende do tipo de arquivo a ser manipulado. O *Apache Flume* superou em 19% o *Apache Kafka* ao trabalhar com arquivos pequenos (tamanho médio de 11,8 KB). Além disso, o *Apache Flume* também foi 10% melhor que seu concorrente ao manipular arquivos grandes (com tamanho médio de 1009 MB). Entretanto, quando se trata de *streaming* de dados em tempo real, a situação se inverte, sendo o *Apache Kafka* 21,6% mais rápido que seu rival.

Este artigo está organizado como segue. A Seção 2 apresenta os trabalhos relacionados. A Seção 3 descreve a avaliação experimental de desempenho dos *brokers*. A Seção 4 detalha os experimentos. A Seção 5 conclui o artigo.

2. Trabalhos Relacionados

Uma arquitetura *big data* geralmente compreende três fases principais, a fase de extração, transferência e processamento de dados [Pääkkönen and Pakkala 2015]. A fase de extração emprega técnicas de coleta de dados em bases de dados. A fase de transferência é responsável por enviar os dados coletados para tratamento e a fase de processamento realiza as análises dos dados com um determinado fim. Os principais trabalhos da literatura sobre avaliação de desempenho de *brokers* se concentram em soluções proprietárias, na fase de processamento de dados ou se concentram em contextos diferentes do *big data*.

Henjes *et. al.* investigaram o desempenho de vazão da solução proprietária *Message Broker WebSphereMQ* [Henjes et al. 2006]. O estudo considerou diferentes números de *subscribers*, *publishers*, mensagens de diferentes tamanhos, diferentes tipos de filtros e estes filtros de diferentes complexidades. Os resultados obtidos revelaram que o tamanho da mensagem tem um impacto significativo sobre a vazão de dados no servidor. Entretanto, a avaliação foi conduzida com apenas um *broker*, o *WebsphereMQ* da IBM, o qual consiste em um software de código fechado.

Córdova avaliou duas das plataformas de código aberto na área de processamento de *big data*, *Storm* e *Spark* [Córdova 2014]. O objetivo deste estudo consistiu em fornecer uma visão geral de muitas características de alto nível de ambos os sistemas, para poder compreender e avaliar as suas velocidades de processamento. Os resultados obtidos revelaram que o *Storm* foi em torno de 40% mais rápido em relação ao *Spark* ao manipular registros pequenos. No entanto, com o aumento do tamanho dos registros o *Spark* conseguiu melhorar seu desempenho, chegando a superar o *Storm*. Todavia, o estudo foi conduzido para analisar a fase de processamento de dados dos *broker*, ignorando as etapas de extração e transferência de dados.

Ionescu avaliou os *brokers* de código aberto *RabbitMQ* e *ActiveMQ* em situações de aumento do tamanho da mensagem no escopo de tecnologias de *middleware* orientados a mensagens [Ionescu 2015]. As métricas analisadas consistiram na velocidade de processamento para envio/recebimento de mensagens e na carga de memória. Os resultados obtidos revelam que o *broker ActiveMQ* possui um desempenho superior quando se trata de receber mensagens, em contrapartida o *RabbitMQ* é mais rápido quando a tarefa a ser executada é a de enviar mensagens para uma aplicação cliente. Todavia, o escopo da pesquisa foi restrito ao contexto de tecnologias de *middleware* orientados a mensagens, não compreendendo o contexto de *big data*.

Este trabalho apresenta uma avaliação experimental de desempenho dos *brokers Apache Flume* e *Kafka* no contexto de *big data*. Assim como em [Henjes et al. 2006], a avaliação proposta analisa a métrica de vazão de dados. No entanto, este trabalho considera apenas soluções de código aberto que podem ser livremente adaptadas para fins específicos. A decisão de avaliar soluções livres corrobora com os objetivos apresentados em [Córdova 2014] e [Ionescu 2015]. Todavia, diferentemente de [Córdova 2014], a avaliação apresentada analisa a fase de extração e transferência de dados e ao contrário de [Ionescu 2015], o trabalho proposto se concentra no contexto de *big data*.

3. Avaliação Experimental de Desempenho dos *Brokers*

Esta seção descreve o projeto de avaliação experimental de desempenho dos *brokers*. A Subseção 3.1 apresenta o projeto do protótipo. A Subseção 3.2 detalha as carga de

trabalho de trabalho utilizadas.

3.1. Projeto do Protótipo

Esta subseção descreve o projeto de um protótipo para interagir com os *broker Apache Flume* e *Kafka*. A Figura 1 mostra como o protótipo interage com o *broker Apache Flume*. Os dados gerados são coletados por um ou mais agentes do *Flume*, sendo que cada agente pode possuir uma ou mais *sources, channels e sinks*. O *source* é responsável por coletar os dados gerados pelos *scripts* e pela aplicação e armazenar no canal configurado. O canal utilizado atua como a memória principal, por ser mais rápida que as outras possibilidades oferecidas pelo *broker*. Quando terminado o transporte ao longo do flume estes dados foram armazenados no sistema de arquivos do *Hadoop* (uma estrutura voltada para ambientes distribuídos que permite realizar análises e armazenamento de grandes conjuntos de dados estruturados e não estruturados), através da configuração do *Sink*. O *Hadoop* é necessário para que se consiga implementar o ambiente esperado para a realização da análise de desempenho entre os *brokers*.

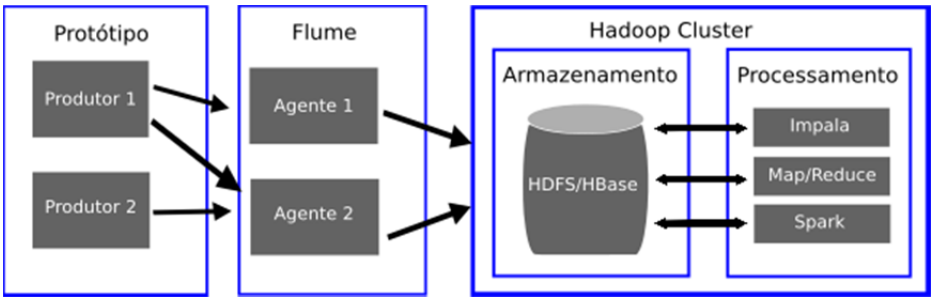


Figura 1. Esquema de interação entre o protótipo, Apache Flume e Hadoop

A Figura 2 apresenta como o protótipo interage com o *Kafka*. O *broker Apache Kafka* não é capaz de realizar a coleta dos dados gerados pelos *scripts* e pela aplicação. Para contornar esta dificuldade, o protótipo produz os dados e alimenta o *broker* enviando mensagens para uma ou mais partições definidas previamente.

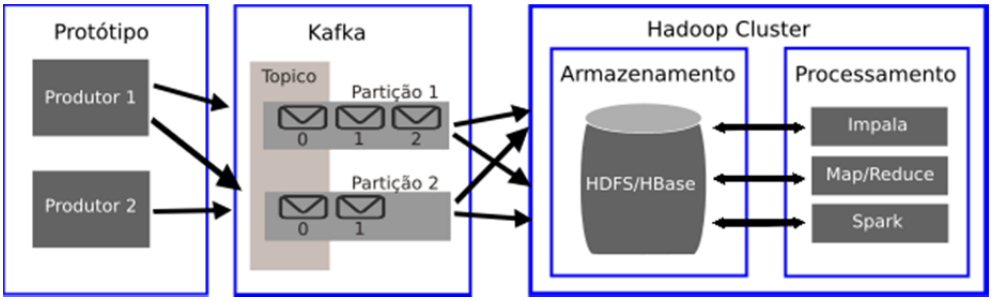


Figura 2. Esquema de interação entre o protótipo, Apache Kafka e Hadoop

3.2. Carga de Trabalho

No contexto de *Big Data*, os dados podem ser adquiridos pro meio de arquivos de *log*, base de dados ou *streaming*, sendo fundamental o uso de *brokers* para a coleta destes dados, para posterior processamento, análise e armazenamento. Para alcançar esta características, a avaliação foi conduzida com dois tipos de carga, uma carga de trabalho artificial e uma carga de trabalho real. A carga de trabalho artificial é produzida através de *shell scripts* desenvolvidos para criar os seguintes conjuntos de dados: (*i*) muitos

arquivos pequenos; (*ii*) poucos arquivos extremamente grandes e (*iii*) uma mescla de muitos arquivos pequenos e grandes. A geração dos arquivos é feita a partir do arquivo *urandom*, o qual é capaz de fornecer caracteres de forma aleatória e ilimitada, sendo nativo nas distribuição Linux.

A carga de trabalho real consistiu em fluxos de dados *streaming*. Para utilizar esta carga de trabalho uma aplicação foi desenvolvida para usar dados reais do *Twitter*. Esta aplicação recupera *tweets* públicos, filtra os *tweets* de acordo com parâmetros informados, agrega essas informações e as formata de acordo com um padrão aceito pelos *brokers*. A Figura 3 apresenta o esquema de funcionamento da aplicação desenvolvida.

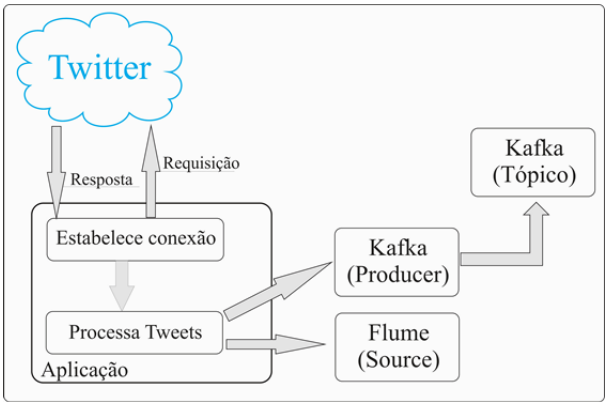


Figura 3. Esquema de funcionamento da aplicação

A aplicação utiliza a API (Interface de Programação de Aplicação) *Twitter streaming API*, a qual possibilita recuperar qualquer *tweet* público. Esta mesma API foi utilizada por [Córdova 2014] para realizar uma análise de desempenho entre dois serviços pertencentes a fase de processamento de dados. Além disso, a aplicação gera duas *Consumer Key*, sendo uma delas secreta e dois *Tokens* de acesso, sendo um deles secreto também. A *Consumer Key* é utilizada para a autenticação do usuário cadastrado no *Twitter*, para conseguir acesso a API *Twitter Streaming*. Os *Tokens* possui a função de autenticar as requisições enviadas ao *Twiter* para a coleta das mensagens [1].

Para o *Kafka* a aplicação foi integrada diretamente com o *driver* produtor do *broker* e do produtor para o tópico configurado. O tópico pode ser definido como uma categoria de mensagens, que possuirá apenas um configurado para os sete *brokers* do *cluster*. Já para o *Flume* basta a correta configuração dos três parâmetros principais, o *source*, *channel* e o *sink* para que seja possível seu funcionamento. Esta aplicação foi responsável por gerar o *streaming* de dados, em tempo real, para a realização dos experimentos.

4. Experimentos

Esta seção apresenta os experimentos realizados e os resultados obtidos. Os experimentos foram conduzidos como objetivo de mensurar o desempenho dos dois *brokers*, o *Apache Kafka* e o *Apache Flume*. A Subseção 4.1 descreve o ambiente utilizado, os cenários de avaliação e as métricas utilizadas. A Subseção 4.2 detalha os resultados obtidos.

[1]https://dev.twitter.com/docs/

4.1. Ambiente, Cenários e Métricas

Esta subseção descreve o ambiente, os cenários e métricas da avaliação de desempenho. Os experimentos foram conduzidos em um ambiente de *cluster* composto por sete computadores *Dell Optiplex GX270*. Cada computador foi equipado com um processador *Pentium 4 HT* 2.80 GHz, 3 GB de memória RAM e 160 GB de armazenamento. O sistema operacional utilizado nestes computadores consistiu na distribuição Linux Rocks 6.1 *Emerald Boa*, projetada para utilização de em *clusters* de alto desempenho. A conexão entre os computadores do *cluster* foi feita por meio de interfaces de rede de 1000 Mbps. A utilização de um *cluster* foi necessária para minimizar qualquer limitação física de processamento. Os experimentos foram conduzidos com as versões 0.8.2.2 do *Apache Kafka*, 3.4.8 do *Apache Zookeeper* e 1.6.0 do *Apache Flume*.

Quatro cenários foram construídos para avaliar os *brokers* em diferentes situações. O primeiro cenário (*Experimento 1*) simula a transferência de arquivos de *logs* de servidores, sendo configurado com uma grande quantidade de arquivos criados artificialmente com tamanho médio de 11,8 KB. O segundo cenário (*Experimento 2*) considera uma situação onde os *brokers* precisariam manipular grandes arquivos, onde foram utilizados arquivos criados artificialmente com tamanho médio de 1009 MB. O terceiro cenário (*Experimento 3*) simula a transferência de arquivos comuns gerados por usuários, com tamanho similar a arquivos de fotos, músicas, mensagens de *email* e documentos de texto. Para alcançar este objetivo, o *Experimento 3* foi projetado para manipular uma mescla de arquivos pequenos e grandes artificialmente criados com tamanho médio 4,6 MB, variando de 10 KB até 10 MB. O quarto cenário (*Experimento 4*) considera uma situação de carga de trabalho real coletada em tempo real do *Twitter*.

Tabela 1. Configuração dos Experimentos

Cenário	Nº de Arquivos	Nº de Brokers	Tam. Total
Experimento 1	2.650	7	31,32 GB
Experimento 2	28	7	28,26 GB
Experimento 3	8323	7	38,31 GB
Experimento 4	N/A	3	N/A

Como mostra a tabela, o experimento 4 não possui um número fixo de arquivos, isso se dá pelo fato que a aplicação coleta fluxo de dados, por isso também, o tamanho total transportado varia para cada um dos *brokers* após as quatro horas de execução. Outro ponto à ser observado está ligado aos três primeiros experimentos, estes utilizando os arquivos artificialmente gerados pelo script, não possuem um tempo de execução fixo, pois variam para cada um dos *brokers*.

A avaliação de desempenho foi conduzida observando duas métricas, a vazão de dados e a taxa de processamento. A análise da vazão de dados possibilita avaliar a quantidade de *bytes* por segundo que um *broker* consegue transportar, quanto maior for a vazão, melhor será o desempenho *broker*. A taxa de processamento apresenta o custo de execução do *broker* para o processador da máquina em que ele está sendo executado, quanto menor o valor desta métrica, melhor o desempenho do *broker*.

4.2. Resultados

Esta subseção detalha os resultados dos experimentos em termos das métricas de vazão e consumo de processamento. Os resultados de vazão mostram o *broker Apache Flume*

superando o *broker Apache Kafka* ao tratar apenas arquivos pequenos e apenas arquivos grandes. A Figura 4, mostra a vazão de dados utilizando arquivos sintéticos.

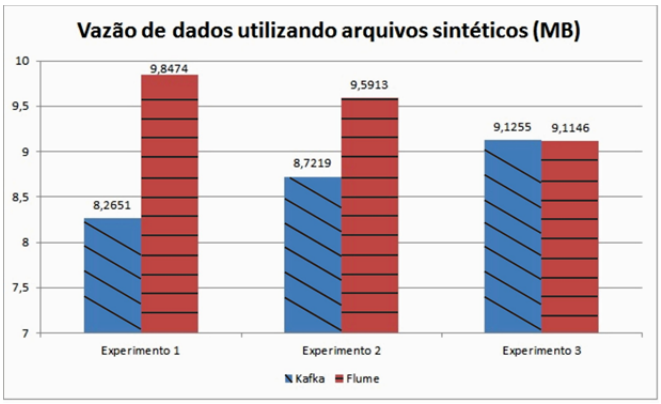


Figura 4. Vazão de Dados Utilizando Arquivos Sintéticos

Os resultados obtidos mostram uma vantagem do *Apache Flume* em dois dos cenários e um resultado próximo ao seu concorrente em apenas uma das execuções. No cenário *experimento 1*, o *Apache Flume* processou 2650578 arquivos apresentando a vantagem de de 19% em relação ao *Apache Kafka*. No cenário *experimento 2*, o *Apache Flume* apresentou uma vazão de dados de 870 KB/s, superando em 10% o seu concorrente. No cenário *experimento 3* ambos os *brokers* tiveram desempenho praticamente igual, sendo o *Apache Kafka* apenas 0,11% ou 10,9 KB/s.

A Figura 5 apresenta os resultados da métrica de vazão do cenário *experimento 4*, a qual utiliza a aplicação desenvolvida. Esta aplicação coletou postagens no *Twitter* em tempo real e entregou aos dois *brokers* sem nenhum filtro de consulta, ou seja, foi coletado qualquer *tweet* público disponível. O experimento teve duração de quatro horas para cada um dos *brokers*. Os resultados coletados apresentam uma vazão de 21,6% ou 126,72 KB/s superior do *Apache Kafka* em relação ao *Apache Flume*.

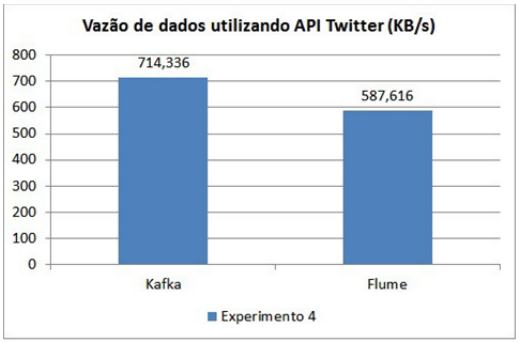


Figura 5. Vazão de dados utilizando a aplicação desenvolvida

Os resultados da métrica de consumo de processamento mostram que os dois *brokers* obtiveram resultados semelhantes. Nos experimentos utilizando a aplicação, a porcentagem de uso do processador, para cada um dos nós, se manteve próximo de 5%, já para os testes utilizando arquivos sintéticos, o *Apache Flume* manteve uma média de 10% durante a execução, já o *Kafka*, demonstrou uma economia de 5% em relação ao seu concorrente. Durante a execução dos experimentos, não foi detectada nenhuma falha no decorrer do transporte dos dados.

Por fim, observando os resultados obtidos, é possível relacioná-los a algumas características de cada *broker*, por exemplo, o *Kafka* tem como foco o processamento de *streaming* de dados em tempo real. A API *producer* contida nele, permite que aplicações desenvolvidas enviem fluxo de dados para os seus tópicos de maneira eficiente. Isso explica o seu melhor desempenho no quarto experimento, utilizando a aplicação desenvolvida. Já nos experimentos anteriores, o Flume obteve um desempenho superior. O principal motivo para isso, se dá pelo fato de que sua arquitetura mais simples ter como foco o processamento de arquivos de *logs* de servidores, através da coleta e agregação destes dados. Por isso este *broker* não depende de nenhum outro *software* para realizar esta coleta, bastando apenas a correta configuração do *source*.

5. Conclusões

Este artigo apresentou uma avaliação de desempenho dos *brokers Apache Flume* e *Kafka* no contexto de *big data*. Foram definidos quatro cenários de avaliação utilizando dados gerados artificialmente e dados reais coletados em tempo real. Duas métricas foram consideradas, a vazão de dados e a taxa de processamento. Os resultados obtidos mostram o *Apache Flume* superando em 19% o *Apache Kafka* ao trabalhar com arquivos pequenos (tamanho médio de 11,8 KB). Além disso, o *Apache Flume* também foi 10% melhor que seu concorrente ao manipular arquivos grandes (com tamanho médio de 1009 MB). Entretanto, quando se trata de *streaming* de dados em tempo real, a situação se inverte, sendo o *Apache Kafka* 21,6% mais rápido que seu rival. Estes resultados vão de encontro com o foco de aplicação de cada *broker*, onde o *Apache Flume* tem sua arquitetura desenvolvida visando um melhor desempenho na coleta e agregação de *logs* de servidores enquanto o *Apache Kafka* possui seu desenvolvimento voltado para o transporte de *streaming* de dados em tempo real. Como trabalho futuro, pretende-se realizar uma análise de desempenho em serviços responsáveis por fazer a análise dos dados coletados.

Referências

Córdova, P. (2014). Analysis of real time stream processing systems considering latency.

Gantz, J. and Reinsel, D. (2011). Extracting value from chaos. *IDC iview*, 1142:1–12.

Gantz, J. and Reinsel, D. (2012). The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east. *IDC iView*, 2007:1–16.

Henjes, R., Menth, M., and Zepfel, C. (2006). Throughput performance of java messaging services using WebsphereMQ. In *IEEE International Conference on Distributed Computing Systems Workshops*, pages 26–26.

Ionescu, V. M. (2015). The Analysis of the Performance of RabbitMQ and ActiveMQ. In *IEEE International Conference-Networking in Education and Research*, pages 132–137.

Pääkkönen, P. and Pakkala, D. (2015). Reference Architecture and Classification of Technologies, Products and Services for Big Data Systems. *Big Data Research*, 2(4):166 – 186.

Ramachandran, U., Modahl, M., Bagrak, I., Wolenetz, M., Lillethun, D., Liu, B., Kim, J., Hutto, P., and Jain, R. (2005). MediaBroker: A pervasive computing infrastructure for adaptive transformation and sharing of stream data. *Pervasive and Mobile Computing*, 1(2):257 – 276.

Uma solução de confirmação para autenticação em ambientes ubíquos baseada na localização do usuário

Luis A. Silva[1], Douglas A. dos Santos[1], Rudimar L. S. Dazzi[2], Valderi R. Q. Leithardt[1]

Laboratório de Sistemas Embarcados e Distribuídos – LEDS[1]
Laboratório de Inteligência Aplicada – LIA[2]
Universidade do Vale do Itajaí (UNIVALI)
Caixa Postal 360 – CEP 88302-901 – Itajaí – SC – Brasil

{luis.silva, douglasas}@edu.univali.br, {rudimar, valderi}@univali.br

Abstract. *In ubiquitous environments, intelligent objects are interconnected and being part of our everyday life. The smart objects tend to increase with the Internet of Things. However, the adaptation of the environment to the user profile and its characteristics has in turn the need to determine its positioning accuracy. This work proposal a authentication and verification of the location and confirmation of the user in the environment. We also present comparisons between technologies such as Wi-Fi, GPS and Bluetooth to confirm location in ubiquitous environments. Experiments and results show a gain from the use of Bluetooth Low Energy (BLE) technology, derived from Bluetooth, normally used for indoor environments.*

Resumo. *Em ambientes ubíquos, objetos inteligentes estão interligados de tal forma a fazer parte do dia-a-dia. Tais objetos inteligentes tendem a aumentar com a Internet das Coisas. No entanto, a adaptação do ambiente ao perfil do usuário e suas características tem por sua vez a necessidade de determinar sua localização com precisão. Desta forma, este trabalho apresenta uma proposta de autenticação e verificação da localização e confirmação do usuário no ambiente. Também apresentamos comparações entre tecnologias como Wi-Fi, GPS e Bluetooth para confirmação da localização em ambientes ubíquos. Experimentos e resultados obtidos demonstram ganho com uso da tecnologia Bluetooth Low Energy (BLE), advinda do Bluetooth, utilizada normalmente para ambientes internos.*

1. Introdução

Os ambientes ubíquos em conjunto com implementações relacionadas ao conceito de Internet das Coisas estão cada vez mais presentes no cotidiano [González-Jaramillo, 2016]. Tais características podem ser relacionadas diretamente ao gerenciamento de perfis de usuário. Estes perfis e dados são utilizados como entrada no intuito de estimular reações e alterações nas condições atuais do usuário, além do ambiente no qual esta inserido o dispositivo utilizado, sendo considerado também o hardware, software, e a adaptação entre eles.

Segundo Wei e Chan (2011), criar regras para tratar adaptações gera uma carga de trabalho muito grande para os desenvolvedores, os quais sentem dificuldades em abordar todas as possibilidades em ambientes altamente dinâmicos. Portanto, é necessário controlar entre outras premissas, a proximidade dos sensores em tempo de execução,

priorizando a localização do usuário, conforme Leithardt et al. (2016). No que diz respeito aos sistemas que utilizam somente o GPS como base para obter a localização do usuário, esses não garantem total precisão em ambientes indoor. Para tanto, foi desenvolvido um modelo de autenticação para ambientes ubíquos com base na localização, visando aumentar a precisão e confiabilidade dos dados obtidos. Contudo, é um desafio obter a localização exata do usuário em ambientes indoor. Em alternativa, existem tecnologias com potencial disponíveis para posicionamento em ambientes internos, sendo que algumas técnicas de localização utilizadas se baseiam em medir a intensidade do sinal recebido. Uma tecnologia candidata é a recente introdução do Bluetooth 4.0, conhecido comercialmente como BLE [Bluetooth, 2010]. O BLE foi desenvolvido para a comunicação entre dispositivos implementados para Internet das Coisas [Faragher e Harle, 2015].

Normalmente um dispositivo BLE é pequeno e comercialmente é conhecido como uma etiqueta, em função do seu formato. Todavia nenhuma das tecnologias possibilita uma completa solução, cada uma com suas vantagens e desvantagens. O método baseado no exemplo faz uso de um sofisticado modelo de geometria para estimar a localização de dispositivos, enquanto o método baseado em impressões digitais, explora técnicas de mineração de dados para recuperar locais a partir de uma série de dados históricos [He e Chan, 2016]. Com base na literatura pesquisada e no modelo proposto, organizamos este trabalho para um melhor entendimento nas seguintes seções: Seção 1 a Introdução, na sequência, a seção 2 apresenta os trabalhos relacionados com uma tabela comparativa entre a solução proposta e os trabalhos pesquisados. A seção 3 apresenta o modelo proposto e a descrição do protótipo desenvolvido. Na seção 4 são apresentados os testes realizados e resultados preliminares. E por fim, apresentamos as conclusões e trabalhos futuros.

2. Trabalhos Relacionados

As técnicas tradicionais de autenticação em ambientes requerem esforços adicionais dos usuários, como por exemplo a utilização de senhas para liberação de acesso. Tal forma, exige abordagens alternativas de autenticação ou ainda aprimoramento de métodos tradicionais. O trabalho de Nunes (2013) proporciona um estudo em relação a tecnologia *Near Field Communication* (NFC), tecnologia esta que ganhou atenção após ser aplicada por fabricantes de smartphones para ser utilizado como provedor de pagamentos, sendo assim, uma alternativa ao uso de cartões magnéticos, o qual efetuam a transação apenas com a aproximação do *smartphone* sobre o dispositivo do estabelecimento.

No trabalho de Liu et al. (2014) foi desenvolvida uma técnica de estimativa de proximidade utilizando a indicação da intensidade do sinal recebido - *Received Signal Strength Indication* (RSSI), definido por uma medida relativa da força da energia de um sinal de rádio utilizando a tecnologia BLE. Foi utilizado como base para realização da classificação empírica para diferentes tipos de distâncias. Sendo a principal contribuição do trabalho, medir a proximidade entre dois dispositivos de usuários distintos, a ponto de revelar se os dois usuários estão próximos uns dos outros ou não. Outros trabalhos demonstram o desenvolvimento de aplicações para dispositivos móveis para interagir com sensores, por exemplo Diaz (2010) e Hansen et al. (2013). Já Faragher e Harle (2015) realizam estudos em ambientes com uma alta densidade de dispositivos BLE e demonstram sua comparação com o Wi-Fi. O estudo de Hansen et al. (2013) é fundamentado em uma aplicação para dispositivos móveis demonstrando uma

triangulação da intensidade do sinal do Wi-Fi, indicando a posição atual do dispositivo em uma planta baixa referente ao local. O trabalho de Ashibani et al. (2017) contribuição relacionada à ciência de contexto, utilizando a localização por meio de Bluetooth, perfil de usuário, calendários, entre outros, para o gerenciamento de uma *smarthouse*. Com base na literatura pesquisada apresentamos a tabela 1 que relaciona os trabalhos pesquisados, comparando as tecnologias utilizadas. Desta maneira é descrita uma comparação dos trabalhos relacionados em relação ao modelo proposto, destacando o fato de este trabalho utilizar mais de uma tecnologia. Os trabalhos relacionados destacam o fato das tecnologias voltadas para navegação e posicionamento em ambientes internos ou externos, possuírem contribuições de pesquisa a serem desenvolvidas.

Tabela 1. Comparação trabalhos relacionados x Solução proposta.

	GPS	Wi-Fi	NFC	Bluetooth	RFID
[Diaz 2010]				X	
[Nunes 2013]			X		
[Hansen et al. 2013]		X			
[Liu et al. 2014]				X	
[Faragher e Harle 2015]				X	
[Leithardt et al. 2016]					X
[Ashibani et al. 2017]				X	
Modelo proposto	X			X	

3. Desenvolvimento

A utilização de um mecanismo de confirmação de autenticação em ambientes deve atender as características do cenário apresentado. Para efetuar todos os processos necessários, este modelo é composto por uma aplicação cliente, disponibilizada no dispositivo do usuário, bem como, um módulo de *middleware* aplicado ao UbiPri [Leithardt et al. 2016] para gerenciamento de dados e requisições. O middleware retorna os dados referentes ao ambiente, como o polígono geométrico com as coordenadas referente ao local, nome, identificador e nível de acesso. Este trabalho não aborda assuntos relacionados a segurança do ambiente, apenas prioriza a identificação do local.

Ao aproximar-se do ambiente juntamente com o dispositivo móvel do usuário, estando o serviço de localização habilitado, faz-se uma varredura, a fim de procurar se naquele ambiente existe algum sensor BLE. Caso encontre um ou mais sensores é feito então uma requisição a aplicação servidor conforme o esquema da Figura 1. O servidor responde com os dados do ambiente em que o dispositivo se encontra, retornando informações referentes ao sensor Bluetooth, como o *Universal Unique Identifier* (UUID). O usuário deve ter instalado em seu dispositivo a aplicação cliente, não sendo necessário estar sempre em primeiro plano, sendo possivel operar em modo *background*, juntamente com seu Bluetooth habilitado.

A Figura 1 ilustra o modelo proposto, incluindo detalhes de quais parâmetros serão coletados e retornados. Entre eles estão os valores incluídos e disponibilizados pelo sensor Bluetooth como o UUID, *Major e Minor*. A combinação destes parâmetros quando enviado ao servidor faz com que se identifique qual sensor está mais próximo e qual ambiente o usuário está localizado. A aplicação cliente desenvolvida manteve como preocupação disponibilizar funcionalidades tanto para a plataforma *Android* como para a plataforma *iOS*, contribuindo assim com a fase de testes em diferentes plataformas.

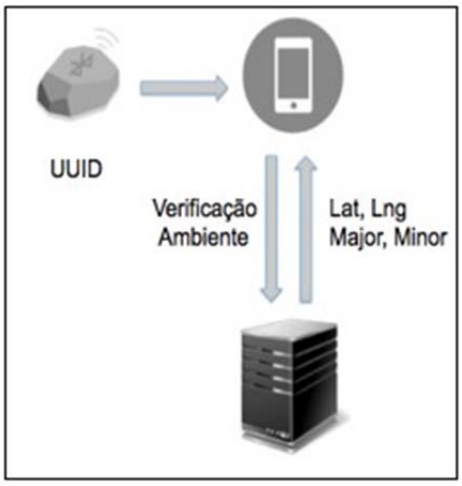


Figura 1. Modelo proposto.

A partir da inicialização da aplicação, é feito então a varredura através do Bluetooth, seguindo intervalos de 10 segundos. Tal intervalo foi configurado a fim de obter melhor economia de bateria do dispositivo utilizado. A varredura em questão, tem por finalidade explorar o ambiente em busca de sensores, identificando principalmente a intensidade de sinal de cada sensor, quando encontrado. Este valor é utilizado para calcular a distância em relação ao sensor. O cálculo utilizado para obter a distância é apresentado na seção 4.

A primeira versão da aplicação apresenta basicamente o valor do identificador único (UUID), o calculo da distância em metros, o ambiente ao qual o sensor pertence, a intensidade do sinal (RSSI) além do campo *Major*, utilizado para mapear um *beacon* dentro de um grupo de *beacons* e o campo *Minor*, utilizado para demarcar as diferenças dentro de um determinado grupo de *beacons*. Assim quando a aplicação requisitar ao servidor, terá a localização como retorno, e a qual ambiente aquele sensor pertence, conforme a Figura 2.

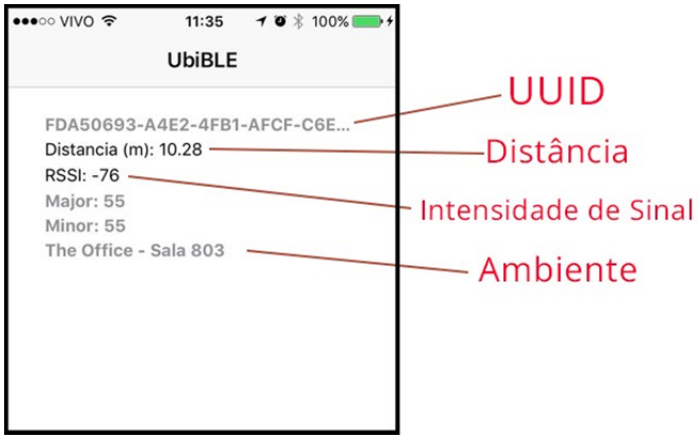


Figura 2. Aplicação desenvolvida para testes.

Um dispositivo BLE opera na mesma frequência de redes sem fio, 2.4GHz e tem como principal característica o baixo consumo de energia. Sua distância máxima de alcance é de até 30 metros, com uma precisão de 1 metro. Além de o dispositivo contar com um identificador único de 16 *bytes*, conta com duas informações nomeadas de *major number* e *minor number*, com capacidade de 2 bytes cada, podendo ser customizado

manualmente com qualquer número dentro de sua capacidade. Por conta de não haver necessidade de fios, pode ser facilmente aplicado em diversos locais ou objetos. De acordo com Zhao e Xiao (2014) ao conduzir uma extensa pesquisa em ambientes indoor e comparar a tecnologia BLE com o método de obter a posição via Wi-Fi, constatou-se uma precisão de 27% a mais do que quando utilizado BLE. A Tabela 2 apresenta um comparativo entre as tecnologias, elencando precisão, custo e consumo de energia. Este ultimo deve-se ter atenção, por se tratar de uma pesquisa abordando o uso de dispositivos dependentes de bateria.

Tabela 2. Comparação entre tecnologia e a precisão. Adaptado de Liu et al. (2014)

Tecnologia	Precisão (m)	Custo (HW)	Cobertura	Consumo de energia
Wi-Fi	3 a 30	Alto	Alta (Indoor)	Alto
GPS	5 a 50	Alto	Alta (Outdoor)	Alto
BLE	1 a 4	Médio	Alta	Médio

A medição da distância do dispositivo até o sensor BLE é feita a partir de um método implementado por Kotanen et al. (2003) e referenciado por Liu et al. (2014), onde é aplicada uma fórmula para obter-se a distância real em metros a partir da intensidade do sinal capturado, obtido pela antena do Bluetooth presente no dispositivo do usuário. Abaixo a representação desta fórmula.

$$RSSI = P_{\&'} + G_{\&'} + G_{+'} + \left(\frac{c}{4\pi f}\right) - 10n \log(d)$$

$$20log$$

$$= P_{\&'} + G - 40.2 - 10n \log(d)$$

$$distancia = 10^{[(\quad)\quad]}$$

Figura 3. Distância baseado no RSSI. Fonte: Adaptado de Kotanen et al. (2003).

A fórmula representada na figura 3 demonstra a obtenção da intensidade do sinal – RSSI, onde G_{tx} e G_{rx} são o ganho da antena e G a soma entre eles, sendo *c* a velocidade da luz ($3.0 * 10^8$ m/s), *f* a frequência operacional do Bluetooth 2.4GHz e *n* o fator de atenuação. Assim, derivando a fórmula poderá ser obtida a distância.

4. Testes e Resultados preliminares

Para fazer a validação do modelo proposto utilizando o BLE como sistema de posicionamento, há alguns pontos a serem considerados, como o algoritmo de transformação de intensidade do sinal em distância, a taxa de transmissão, orientação da antena bem como o local onde são feitos os testes, além da quantidade de dispositivos testados.

Os testes foram realizados em um apartamento com aproximadamente $90m^2$ em um dia normal, com diversos aparelhos eletrônicos distribuídos pelo ambiente, como um roteador Wi-Fi e um telefone sem fio, ambos utilizando a frenquencia 2.4GHz. A figura 4 representa o ambiente, situando a localização dos sensores Bluetooth e do ponto de acesso Wi-Fi. Os sensores BLE foram colocados a uma altura média de 1,70m, a fim de garantir um campo aberto na vertical, entre o dispositivo emissor até o receptor do sinal.

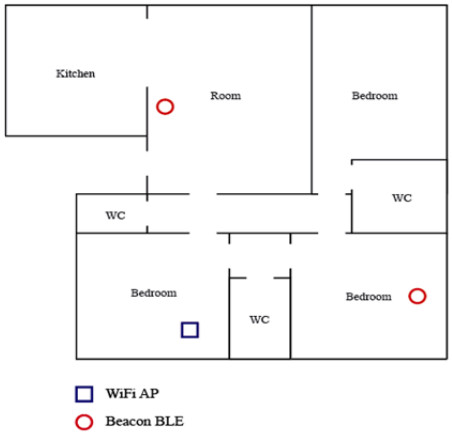


Figura 4. Ambiente para testes, localização dos sensores BLE e acesso WiFi.

Durante os testes foram utilizados dois smartphones de diferentes arquiteturas. Um modelo iPhone 5S com sistema operacional *iOS* e outro modelo Motorola X Play com sistema operacional *Android*. Foram utilizados dois *beacons* BLE *April Brother* embarcados com *system-on-chip* (SoC) CC2450, com distância máxima de 30 metros de operação. O algoritmo fundamenta-se em valores de intensidade de sinal RSSI, sendo assim é importante considerar que estes valores não são constantes, devido ao modo de operação do dispositivo BLE. Este modo conta com 40 canais distintos, cada um com 2 MHz de espaçamento [Hansen et al. 2013].

Os canais estão divididos em dois tipos: *advertising channels* e *data channels*, sendo que os canais de *advertising* ou canais de propaganda são apenas 3 dos 40, deixando os outros 37 apenas para dados. Estes canais normalmente são utilizados para descobrir novos dispositivos, como um celular, e ainda estabelecer uma conexão. Por operar na frequência de 2.4GHz, a mesma utilizada para o WiFi, os dispositivos utilizam a técnica de *Adaptive Frequency Hopping (AFH)*. Este processo faz com que o mapa de frequências disponíveis seja readaptado, fazendo com que os dispositivos não utilizem as frequências já ocupadas. Os testes realizados a fim de obter a precisão em diferentes distâncias em relação ao ambiente foram executados sequenciamente 12 vezes em um período de 24 horas, sendo repetido no dia seguinte o mesmo número de vezes, respeitando sempre as mesmas posições conforme demonstrado na figura 5.

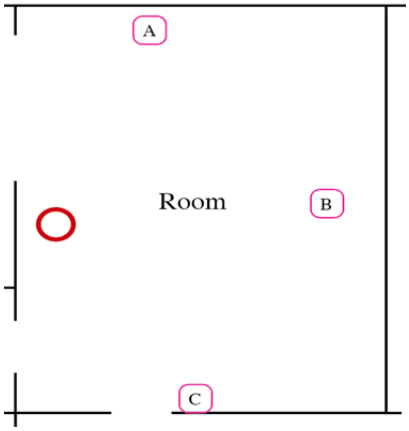


Figura 5. Teste realizado em 3 pontos distintos do ambiente 1.

A Tabela 3 organiza os dados coletados e os resultados dos pontos A, B e C, conendo duas colunas de distâncias, onde o valor da coluna distância 1 foi obtido a partir do modelo demonstrado na Seção 4.3 do artigo, e a distância 2 foi obtida através da aplicação de referência disponibilizada pelo fabricante do sensor BLE.

Tabela 3. Testes realizados no ambiente.

Ponto	Distância 1 (metros)	Distância 2 (metros)	Intensidade de Sinal (dBm)
A	3,2	4,5	-67
B	4,49	5,2	-71
C	5,60	6,5	-74

Um conjunto de testes foi realizado utilizando diferentes medidas entre o *smartphone* e o sensor Bluetooth. Com o dispositivo situado em alternadas posições, dentro de cada uma (para cada distância selecionada) aproximadamente 50 medições foram realizadas. A partir destes 50 valores obtidos, foi adquirida uma média dos valores para representar o valor RSSI. Na Tabela 4, é demonstrada cada distância na seguinte sequencia (1m, 2m, 3m, 4m, 5m, 7.5m, 10m, 15m e 20m). Nota-se que a dispersão é bastante alta. Para esse efeito, calculamos o valor médio do RSSI em dBm e o desvio padrão "(1)" para cada distância aferida, utilizando fórmulas padrão.

Tabela 4. Média de RSSI em relação a distância fixa e desvio padrão

Distância	1m	2m	3m	4m	5m	7.5m	10m	15m
RSSI (dBm)	-58	-70	-70.75	-74.08	-76	-82.89	-85.92	-90.15
Desvio padrão	0.894	2.357	2.220	0.996	3.464	2.201	1.754	1.402

5. Conclusão e Trabalhos Futuros

Os testes realizados neste trabalho confirmaram a possibilidade de utilizar um sensor Bluetooth unido ao GPS para verificar a presença do usuário no ambiente. No entanto, ainda há algumas divergências no sentido de verificar a precisão da posição, sendo necessário a utilização de mais dispositivos em um mesmo ambiente, podendo assim, aferir as distâncias com maior exatidão.

O uso de sensores Bluetooth possibilitou tratar automaticamente a presença de um usuário no ambiente, unindo assim a outros trabalhos na linha de pesquisa, havendo a possibilidade de adaptar o ambiente baseando-se no perfil do usuário, além de tratar envio de alertas e notificações personalizados. Em trabalhos futuros, pretende-se expandir os testes realizados, implantando o sistema de posicionamento BLE em um ambiente universitário contemplando salas de aula, laboratórios, entre outros. Identificar entradas e saídas de alunos em grande quantidade e locais diversos, além de acessos a recursos, auxiliando no gerenciamento do perfil individual. Para tanto, novos algoritmos, testes e resultados deverão ser implementados.

6. Agradecimentos

Os autores agradecem a CAPES pela concessão da bolsa de pesquisa, possibilitando assim a realização do trabalho e os resultados preliminares obtidos. Assim como agradecem ao Laboratório de Sistemas Embarcados e Distribuídos (LEDS) - UNIVALI pelo apoio a pesquisa.

Referências

Ashibani, Yosef, et al. "A context-aware authentication framework for smart homes." Electrical and Computer Engineering (CCECE), 2017 IEEE 30th Canadian Conference on. IEEE, 2017.

Bluetooth, S. I. G. "Bluetooth core specification version 4.0." Specification of the Bluetooth System (2010).

Diaz, Javier JM, et al. "Bluepass: An indoor bluetooth-based localization system for mobile applications." Computers and Communications (ISCC), 2010 IEEE Symposium on. IEEE, 2010.

Faragher, Ramsey and Robert Harle. "Location fingerprinting with bluetooth low energy beacons." IEEE journal on Selected Areas in Communications 33.11 (2015)

Kotanen, Antti, et al. "Experiments on local positioning with Bluetooth." Information Technology: Coding and Computing [Computers and Communications], 2003. Proceedings. ITCC 2003. International Conference on. IEEE, 2003.

Leithardt, V.R.Q. et al. "The classification of algorithms for Privacy Management in Ubiquitous Environments". XXXVI Congresso da Sociedade Brasileira de Computação (CSBC). 8º SBCUP - Simpósio Brasileiro de Computação Ubíqua e Pervasiva, 2016.

Nunes, Bruno Romeu et al. An automation system for ubiquitous computing. 2013. 63f. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2013.

S. He; S. H. G. Chan, "Wi-Fi Fingerprint-Based Indoor Positioning: Recent Advances and Comparisons", IEEE Communications Surveys & Tutorials, vol. 18, no. 1, pp. 466-490, Firstquarter 2016

R. Hansen, et al. "SmartCampusAAU -- An Open Platform Enabling Indoor Positioning and Navigation". IEEE 14th International Conference on Mobile Data Management, Milan, 2013, pp. 33-38.

V. H. González-Jaramillo, "Tutorial: Internet of Things and the upcoming wireless sensor networks related with the use of big data in mapping services; issues of smart cities 2016" Third International Conference on e Democracy & eGovernment (ICEDEG), Sangolqui, 2016, pp. 5-6.

Wei, Edwin JY et al. "CAMPUS: A middleware for automated context-aware adaptation decision making at run time". Pervasive and Mobile Computing, 2011.

Xiaojie Zhao, Zhuoling Xiao, "Does BTLE measure up against WiFi? A comparison of indoor location performance". European Wireless (2014)

Processamento de Sinais Sociais: Visão Geral, Metodologias e Desafios

Isadora V. e Souza[1], William B. Pereira[1], João C. D. Lima[1]

[1]Programa de Pós-Graduação em Informática – Universidade Federal de Santa Maria
Av. Roraima, 1000 – 97.105-900 – Santa Maria – RS – Brasil

isouza@inf.ufsm.br, williambortoluzzipereira@gmail.com,
caio@inf.ufsm.br

Abstract. *Social Signal Processing (SSP) is a field which studies the verbal and nonverbal signals from humans during a social interaction. However, it still has many challenges, such as specific sensors, battery life, privacy and others. With increasing use of mobile devices researchers attention arisen in order to solve those challenges. This article provides a channel for the dissemination of knowledge, since the SSP is a broad area to deepen knowledge about human behavior.*

Resumo. *O Processamento de Sinais Sociais (SSP) é uma área que estuda os sinais verbais e não verbais reproduzidos por seres humanos durante uma interação social. A área ainda possui muitos desafios de pesquisa como: sensores específicos, duração de bateria, privacidade, entre outros. Com o crescente uso de dispositivos móveis, a área chamou a atenção dos pesquisadores que estão trabalhando no desenvolvimento de soluções para alguns desses desafios. Este artigo oferece um canal para a disseminação do conhecimento, pois o SSP é uma área ampla para aprofundar os conhecimentos sobre o comportamento humano.*

1. Introdução

A psicologia é a ciência que estuda o comportamento humano. Alguns métodos utilizados pelos pesquisadores são: questionários, observação, imagens obtidas por câmeras, entre outros. Estes métodos podem conter erros causados pela imparcialidade e a indiferença das pessoas que estão sendo avaliadas, como também por restrições de escalabilidade, pois a avaliação não é feita de modo globalizado, esta avaliação é aplicada em um grupo limitado de pessoas.

Para automatizar o processo de avaliação do comportamento humano, surgiu o *Social Signal Processing* (SSP), ou traduzindo, Processamento de Sinais Sociais que é uma área focada em detectar automaticamente as informações sobre o comportamento social dos usuários, visando a não intrusão, ou seja, utiliza-se o máximo de sensores para levantamento de dados para capturar informação dos contextos que o usuário se encontra. O SSP soluciona os problemas de imparcialidade, indiferença e escalabilidade.

O SSP foi criado não apenas para facilitar o trabalho dos pesquisadores da área de psicologia, mas também para diversas outras áreas, como saúde, engenharia organizacional e marketing, por exemplo. Existem aplicações de monitoramento de frequência cardíaca, pressão, atividade física, outras para medir o nível de satisfação em um ambiente de trabalho e também para ajustar campanhas de determinado produto ou serviço de acordo com o humor e as preferências do usuário.

O artigo está dividido da seguinte forma: na seção 2 é apresentada uma visão geral de Processamento de Sinais Sociais, na seção 3 há um compilado de trabalhos que utilizaram o SSP, na seção 4 os desafios da área e na seção 5 a conclusão.

2. Visão Geral

O Processamento de Sinais Sociais utiliza inúmeros sensores para o levantamento de dados do comportamento do usuário. Segundo [Schuster 2013], o SSP analisa o comportamento humano baseado em pistas verbais e não verbais de baixo nível, como a fala do usuário, piscar e sorrir com o objetivo de detectar comportamentos de alto nível, como atividade física, engajamento e estresse.

Para o SSP é fundamental que a aplicação seja oportunística e não intrusiva, ou seja, que a aplicação levante os dados sem a intervenção do usuário e sem atrapalhar o seu comportamento habitual do cotidiano, quanto maior a naturalidade do processo, maior a legitimidade dos dados adquiridos. Sendo assim, a computação móvel é um dos principais meios de coleta de dados, pois já é uma ferramenta de uso recorrente pelo usuário.

O *SSP Mobile* possui os seguintes objetivos: utilizar abordagens não intrusivas, capturar dados da perspectiva do usuário para produzir dados personalizados, utilizar múltiplas modalidades para extrair informações mais robustas e confiáveis do comportamento, processo de inferência e sensoriamento contínuo, sem restrições de mobilidade e escalabilidade e eliminar a necessidade de hardwares externos.

Vinciarelli (2010) definiu alguns procedimentos para a detecção de comportamento social, que mais tarde foram adaptados para o meio *Mobile* por [Palaghias 2014], que concluíram que para extrair conhecimentos sobre comportamento social em dispositivos móveis são necessários quatro passos:

1. Sensoriamento
2. Detecção de interação social
3. Extração de pistas comportamentais
4. Obtenção de conhecimento sobre comportamento social através de inferência de sinais sociais

O sensoriamento (passo 1) é feito pelos sensores presentes no dispositivo, como giroscópio, acelerômetro, câmera, microfone, etc. Cada sensor é responsável por gerar os dados de determinadas características, como por exemplo o microfone vai gerar os dados referentes à fala do usuário.

A detecção de interação social (passo 2) pode ser realizada de diversas maneiras, pode ser utilizando uma modalidade única ou modalidades múltiplas. Na modalidade única são utilizados conexões Bluetooth ou Wi-Fi para identificar pessoas próximas. Na modalidade múltipla são utilizados conexões Bluetooth e Wi-Fi, microfone, câmera, entre outros sensores. O levantamento de dados de interações sociais possibilitam a extração de pistas comportamentais (passo 3), que são características, hábitos ou padrões de interações do usuário com as pessoas.

Durante uma interação social deve ser levado em conta não apenas o diálogo, mas também os gestos, as feições, o direcionamento da fala, etc. Todos esses atributos são chamados de sinais sociais. Pentland (2008) descreve sinais sociais como sinais de comunicação não-verbal emitidos quando as pessoas estão interagindo socialmente. A

união destes sinais sociais por um período de tempo leva ao conhecimento do comportamento social (passo 4). A Figura 1 apresenta alguns sinais sociais reproduzidos durante uma interação, como: características da postura, olhar mútuo, comportamento verbal, etc.

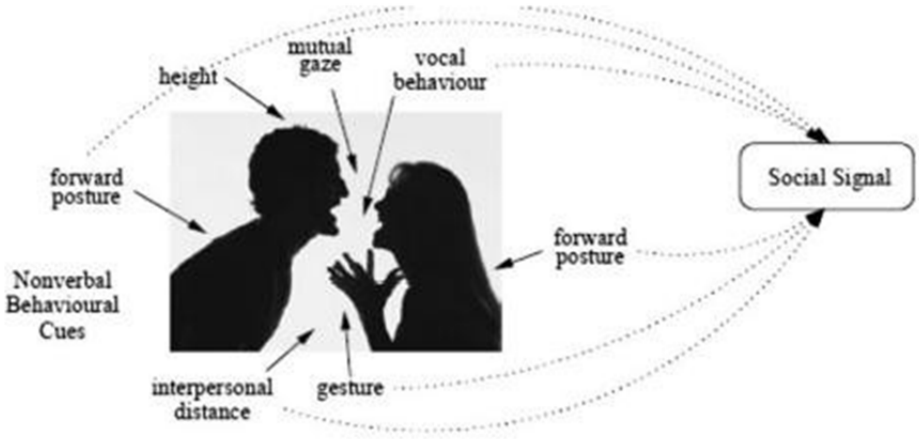


Figura 1. Representação de Sinais Sociais. [Zhou 2010]

A arquitetura do *SSP Mobile* possui quatro bases principais: o framework de sensoriamento, as interações sociais, as pistas comportamentais e os comportamentos sociais inferidos pelos sinais sociais (ver Figura 2). O framework de sensoriamento é necessário para coletar e armazenar dados brutos de sensores ambientais (microfone, câmera), sensores de posição (Bluetooth, GPS, WiFi), sensores virtuais (informação de chamadas, SMS, uso do celular) e inerciais (acelerômetro, giroscópio) e utilizá-los para fazer a inferência de sinais sociais, por exemplo: coletar dados do acelerômetro para inferir que o usuário está realizando determinado gesto.

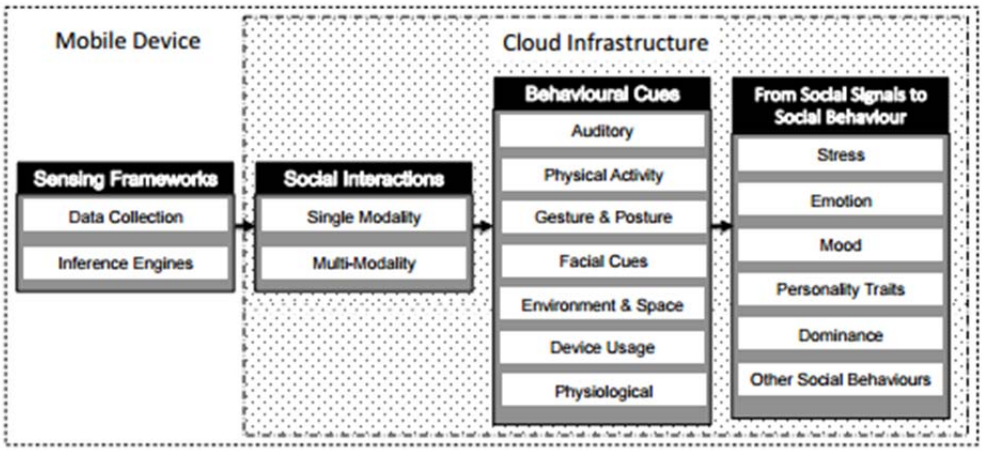


Figura 2. Arquitetura do SSP Mobile. [Palaghias 2014]

As interações sociais são inferidas com base nos sinais sociais previamente levantados pelo framework de sensoriamento. As interações podem ser inferidas por meio de modalidade única ou por modalidade múltipla. Na modalidade única são considerados apenas dados de sensores de posição, principalmente Bluetooth e WiFi, para inferir se as pessoas integrantes da interação estão próximas. Na modalidade múltipla, são

considerados outros tipos de sensores além dos sensores de posição, como sensores ambientais para obter dados do diálogo dos integrantes.

As pistas comportamentais representam informações sobre atividades físicas, fala, gestos, postura, pistas faciais (piscar, sorrir), condições ambientais e espaciais e uso do dispositivo móvel. Essas informações são necessárias para a inferência de características complexas, como: estresse, emoção, humor, traços de personalidade, domínio, entre outras.

3. Exemplos Práticos

O processamento, sensoriamento e monitoramento feito por dispositivos móveis estão naturalmente sendo inseridos na vida dos usuários, abrindo espaço para o desenvolvimento de aplicações que facilitam o cotidiano em várias esferas. Os trabalhos relacionados foram selecionados com o objetivo de demonstrar a abrangência da área de Processamento de Sinais Sociais, destacando sua importância para várias áreas do conhecimento. A Tabela 1 representa um quadro comparativo entre eles.

Tabela 1. Quadro Comparativo dos Trabalhos Relacionados

Sistemas	Sensores	Objetivo	Dados Coletados
SPA	Oxímetro de pulso, medidor de pressão arterial, acelerômetro, temperatura, umidade, luz e GPS	Monitoramento de saúde	Ambientais e biomédicos
BALANCE	MSP box: acelerômetro 3D, barômetro, umidade, luz e GPS	Auxiliar emagrecimento	Calorias consumidas e adquiridas
StudentLife	Acelerômetro, GPS, microfone, proximidade e luz	Pesquisa de saúde mental	Dados do cotidiano do usuário
Social Serendipity	Bluetooth e WiFi	Rede social	Interesses do usuário
Sociômetros	Transmissor de rádio de alta frequência e microfone	Pesquisa de diferença entre gêneros	Discurso e proximidade dos usuários

O SPA de Sha (2008) monitora continuamente o corpo, o comportamento e o ambiente do usuário durante as suas atividades diárias e notifica profissionais de saúde, caso ocorra alguma emergência. A arquitetura do SPA é composta por uma rede de sensores no corpo do usuário para coletar dados ambientais e biomédicos, um servidor remoto que armazena e analisa os dados dos sensores e um grupo de profissionais de saúde que observam os registros feitos pelo SPA e dão conselhos aos usuários para melhorarem suas condições de saúde.

Assim como o SPA, o BALANCE de Denning (2009) também é um aplicativo voltado para a área da saúde, porém tem o objetivo de fornecer uma maneira fácil de

visualizar as calorias consumidas e gastas durante o dia para auxiliar no processo de emagrecimento dos usuários. O aplicativo possui uma base de dados de alimentos com suas devidas calorias e o usuário seleciona os que foram consumidos. O BALANCE utiliza *Mobile Sensing Platform* (MPS) para calcular as calorias gastas em atividades como sentado, andando, correndo e andando de bicicleta.

O StudentLife de Wang (2014) é uma aplicação desenvolvida para pesquisa. Ele faz o sensoriamento contínuo da saúde mental, performance acadêmica e tendências comportamentais dos estudantes. O aplicativo infere atividades como parado, caminhando, correndo, dirigindo e andando de bicicleta e a duração do sono. Além de capturar o áudio e utilizar modelo oculto de Markov para inferir conversações para obter dados da socialização do usuário. Informações sobre o uso do dispositivo móvel também são avaliadas. Os pesquisadores perceberam que estas características influenciam no bem estar e na saúde mental do usuário.

O Social Serendipity de Eagle & Pentland (2005) é uma aplicação que utiliza endereços de hardware Bluetooth e um banco de dados de perfis de usuários que tem o objetivo de viabilizar as interações casuais face a face entre os usuários próximos que não se conhecem, mas deveriam se conhecer pois possuem muitas características em comum. O Serendipity utiliza um servidor centralizado para coordenar as interações sociais locais. O servidor guarda os perfis dos usuários com dados necessários para fazer a combinação destes perfis, como as variáveis de atributos que definem os interesses e o ID do Bluetooth (BTID) do dispositivo. Quando dois usuários do Serendipity se aproximam, o servidor verifica se as pessoas têm interesses e gostos em comum através de uma pontuação de semelhança das variáveis, caso tenham, um sinal é enviado para ambos os dispositivos.

Os Sociômetros de Onnela (2014) são dispositivos portáteis que capturam as pessoas próximas através de um transmissor de rádio de alta frequência e um microfone para acompanhar o discurso dos usuários dentro de contextos específicos. Eles auxiliam nas pesquisas de interação humana, antes feitas através de relatos dos indivíduos. Os auto relatos estão propensos a erros, sendo assim, reduzem a gama de comportamentos e o número de assuntos que podem ser estudados simultaneamente. O objetivo deste estudo com os sociômetros é encontrar algumas diferenças entre os gêneros (masculino e feminino), como loquacidade e estilo de interação em contextos diferentes.

4. Desafios

Com o aumento do mercado de dispositivos móveis nos últimos anos, a área de computação móvel vem ganhando cada vez mais foco de pesquisa. No entanto, ainda existem algumas necessidades e desafios a serem supridos, como: reconhecimento de contexto, precisão de sensores, gerenciamento de bateria, privacidade.

O reconhecimento de contexto é fundamental para o SSP mas também é um grande desafio para a área por ter um amplo sentido do termo. Uma medida que os pesquisadores vem tomando é limitar o âmbito da aplicação, por exemplo: uma aplicação para monitorar a produtividade em determinada empresa.

A tecnologia mais importante para a área de Processamento de Sinais Sociais é o sensor. Atualmente existem inúmeros sensores disponíveis, porém eles são extremamente suscetíveis a erros por não terem o nível de precisão adequado para o sensoriamento de sinais sociais. Por exemplo, o GPS não possui precisão suficiente quando o usuário está dentro de um edifício (vários andares) ou um túnel (sem sinal). Outro ponto, é que ainda

não foram desenvolvidos sensores adequados para detectar os sinais sociais e as atividades do usuário. Para os sinais sociais, estão sendo utilizados sensores invasivos para obter uma melhor acurácia, isso prejudica a naturalidade do sinal. Para as atividades como correndo e andando, estão sendo utilizados alguns sensores como acelerômetro mas ainda com um baixo nível de acurácia.

Outro problema relacionado aos sensores é a fusão de modalidades. O crescente poder computacional dos dispositivos móveis viabilizou a criação de muitos sensores de várias modalidades, o desafio é a união de modalidades de sensores físicos e virtuais mantendo a precisão de inferência de comportamento.

O gerenciamento de bateria é um grande problema da área, pois o SSP utiliza vários sensores e o consumo de bateria nos dispositivos móveis é elevado. As aplicações precisam criar mecanismos para habilitar e desabilitar os sensores de acordo com a necessidade, para não desperdiçar energia. O processamento dos dados dos sensores é, na maioria dos casos, feito em servidores para não consumir energia com processamento. Porém estas medidas não são suficientes em aplicações que precisam detectar várias características no cotidiano do usuário.

A privacidade é também um desafio, pois as aplicações tendem a ser oportunistas, ou seja, capturar dados sem a intervenção do usuário para poder capturar dados de ações realizadas naturalmente. Por isso é preciso ter o consentimento do usuário, garantir o seu anonimato e não enviar dados a aplicativos de terceiros.

5. Conclusão

O Processamento de Sinais Sociais e os dispositivos móveis formaram uma promissora ferramenta para o estudo do comportamento humano, pois os dispositivos são pervasivos, onipresentes e discretos. Porém, há um longo caminho para satisfazer os desafios da área, tanto no desenvolvimento de hardware (sensores, bateria), quanto no desenvolvimento de middlewares e frameworks para facilitar a manipulação de dados em baixo nível. Em conjunto com a psicologia, a mineração e inferência de dados coletados, o SSP tem muito potencial para aprofundar os conhecimentos referentes aos seres humanos.

Referências Bibliográficas

Denning, T., Andrew, A., Chaudhri, R., Hartung, C., Lester, J., Borriello, G., & Duncan, G. (2009). BALANCE: Towards a Usable Pervasive Wellness Application with Accurate Activity Inference. In Proceedings of the 10th workshop on Mobile Computing Systems and Applications. ACM.

Eagle, N., & Pentland, A. (2005). Social Serendipity: Mobilizing Social Software. IEEE Pervasive Computing.

Onnela, J. P., Waber, B. N., Pentland, A., Schnorf, S., & Lazer, D. (2014). Using Sociometers to Quantify Social Interaction Patterns. Scientific reports.

Palaghias, N., Hoseinitabatabaei, S. A., Nati, M., Gluhak, A., & Moessner, K. (2016). A Survey on Mobile Social Signal Processing. ACM Computing Surveys (CSUR).

Pentland, A., & Heibeck, T. (2010). Honest Signals: How They Shape our World. MIT Press.

Schuster, D., Rosi, A., Mamei, M., Springer, T., Endler, M., & Zambonelli, F. (2013). Pervasive Social Context: Taxonomy and Survey. ACM Transactions on Intelligent Systems and Technology (TIST).

Sha, K., Zhan, G., Shi, W., Lumley, M., Wiholm, C., & Arnetz, B. (2008). SPA: A Smartphone Assisted Chronic Illness Self-Management System with Participatory Sensing. In Proceedings of the 2nd International Workshop on Systems and Networking Support for Health Care and Assisted Living Environments. ACM.

Vinciarelli, A., Murray-Smith, R., & Bourlard, H. (2010). Mobile Social Signal Processing Vision and Research Issues. In Proceedings of the 12th international conference on Human computer interaction with mobile devices and services. ACM.

Wang, R., Chen, F., Chen, Z., Li, T., Harari, G., Tignor, S., & Campbell, A. T. (2014). StudentLife: Assessing Mental Health, Academic Performance and Behavioral Trends of College Students Using Smartphones. In Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing. ACM.

Zhou, J., Sun, J., Athukorala, K., Wijekoon, D., & Ylianttila, M. (2012). Pervasive Social Computing: Augmenting Five Facets of Human Intelligence. Journal of Ambient Intelligence and Humanized Computing.

Avaliação de Desempenho Temporal da Comunicação de um Sistema de Controle Baseado em Ethernet

Eduardo Kochenborger Duarte[1], João Cesar Netto[1], João Ricardo Wagner de Moraes[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Av. Bento Gonçalves, 9500 - Porto Alegre, RS - Brasil

{eduardo.duarte,netto,joao.moraes}@inf.ufrgs.br

Abstract. *In some industrial applications, the response time and the determinism are essential, making it necessary for the devices used to have a good performance. There are various factors that can lead to a performance degradation in a controller, like problems in the implementation of the communication with the I/O modules, or scheduling problems. The task of determining and otimizing the bottleneck of a device can be quite complex. This work has the objective of studying the impact in the performance caused by the communication between Nexto series controllers and I/O modules, proposing a methodology to be used as a tool in the otimization process.*

Resumo. *Em algumas aplicações industriais, o tempo de resposta e o determinismo são essenciais, sendo necessário que os dispositivos utilizados possuam desempenho satisfatório. Existem diversos fatores que podem levar à degradação do desempenho de um controlador, como problemas de implementação na comunicação com os módulos de I/O, ou problemas de escalonamento. A tarefa de determinar e otimizar o gargalo de um dispositivo pode ser bastante complexa. Este trabalho tem o objetivo de estudar o impacto no desempenho causado pela comunicação entre controladores da série Nexto e módulos de I/O, propondo uma metodologia para ser utilizada como ferramenta no processo de otimização.*

1. Introdução

A automação de tarefas outrora executadas manualmente é recorrente na indústria e causa grande aumento de produtividade. Por um longo tempo, a automação existiu em escalas bastante diminutas, utilizando-se principalmente de dispositivos mecânicos. Com o surgimento e disseminação da utilização de computadores, a flexibilidade adquirida possibilitou a automatização de praticamente qualquer tarefa [Gupta e Arora 2009].

Um controlador lógico programável (CLP) é um dispositivo que captura informações do ambiente e produz uma reação baseada na forma como foi programado. As informações vêm de sensores, como sensores de temperatura ou pressão, por exemplo. Usando esses dados, o CLP produzirá uma saída de acordo com a lógica programada pelo usuário. Por exemplo, pode-se ativar um atuador que controle um ventilador uma vez que a temperatura suba além de um limiar aceitável.

Existem diversos tipos de aplicações para as quais os CLPs são adequados. Naturalmente, é necessário fazer um levantamento de requisitos específicos à aplicação antes de se avaliar qual equipamento é o mais apropriado (utilizando-se alguma métrica). O

CLP de mercado da série Nexto, disponível no Instituto de Informática, em função de suas características, não possui o desempenho necessário para aplicações com requisitos de desempenho muito altos, considerando o tempo de ciclo como métrica.

Cada tipo de CLP é desenvolvido com o objetivo de se adequar a uma determinada gama de aplicações. A série Nexto é orientada para alta disponibilidade com soluções de redundância e suporte a grandes quantidades de pontos de entrada e saída. Outros equipamentos podem ter mais recursos de hardware, ou serem desenvolvidos para aplicações mais específicas. Um exemplo disso é mostrado na tabela 1 [Série Nexto 2017], [Beckhoff Product Overview 2017], [Beckhoff HW Documentation 2017], [Motion Controllers 2017].

Tabela 1. Comparativo de CLPs

CLP	CLP de Mercado	Tempo de Ciclo	Quantidade de Pontos
Uso geral	Nexto	Ordem de 5 ms	Grande
Uso geral com processador mais robusto	CX	Ordem de 3 ms	Grande
Motion Control	Q-Series	< 1 ms	Muito Pequena

O que se deseja é poder utilizar os controladores Nexto em sistemas de controle com requisitos muito fortes de desempenho. Existem indícios que o desempenho pode ser melhorado através de otimizações na comunicação do CLP com os módulos de I/O. O objetivo deste trabalho é formular uma estratégia para determinar quais os principais pontos passíveis de melhorias, identificando se a comunicação está impactando o desempenho. Para isso, serão apresentados testes que verifiquem o estado da situação atual no que diz respeito a desempenho, bem como uma análise sobre as possíveis causas e soluções.

A seção 2 contextualiza o leitor no que diz respeito à comunicação em ambiente industrial. Na seção 3, são apresentados conceitos básicos sobre CLPs. Em seguida, na seção 4, são apresentadas características, métricas e requisitos para sistemas de tempo real. A seção 5 descreve alguns trabalhos desenvolvidos na área de avaliação de desempenho. O método proposto para avaliar o desempenho da comunicação dos controladores está descrito na seção 6. Por fim, na seção 7, são apresentadas as conclusões do trabalho.

2. Comunicação em Ambiente Industrial

A comunicação através de redes digitais está bem estabelecida em ambientes industriais. Existem diversos sistemas *fieldbus* que são usados em indústrias comerciais, como Modbus e PROFIBUS, por exemplo. No entanto, com a grande disseminação do uso de Ethernet em outras áreas, existem tendências que levam ao uso de protocolos baseados em Ethernet na área industrial. No contexto atual, há uma grande produção de produtos e componentes adaptados a essa tecnologia, diminuindo custos e aumentando o interesse em utilizá-la [Neumann 2007].

No ambiente industrial, existe um conjunto de restrições que exige características não apresentadas pelo Ethernet convencional. Por este motivo, não é possível utilizar essa tecnologia diretamente. Diversos protocolos baseados em Ethernet foram desenvolvidos,

como EtherNet/IP, PROFINET e EtherCAT [Neumann 2007]. No equipamento disponível, a comunicação do controlador com os módulos de I/O possui diversas finalidades (detalhado na seção 6). O protocolo utilizado é o EtherCAT, um protocolo de comunicação industrial de tempo real e alta performance.

O EtherCAT funciona em uma configuração mestre/escravo, onde o mestre envia comandos aos nodos escravos, que podem escrever e/ou ler os dados [OMRON Automation Pvt Ltd 2016]. Através desse mecanismo, a atualização de I/O acontece ciclicamente no tempo. O objetivo é maximizar a utilização do canal de comunicação e minimizar os tempos de resposta.

Os comandos são inseridos no segmento de dados de um telegrama EtherCAT. Um telegrama é composto por um ou mais endereços de escravos, dados e alguns bits de controle. Esses telegramas são inseridos como *payload* em um *frame* Ethernet. Cada um desses *frames* pode conter diversos telegramas EtherCAT.

Um *frame* enviado pelo mestre é repassado apenas para o primeiro nodo escravo, que o repassa ao próximo, e assim sucessivamente. A leitura ou inserção de dados se dá neste momento, quando o *frame* está sendo transmitido, ou seja, não ocorre uma pausa na transmissão: os dados são lidos/inseridos *on the fly*. Uma vez que o *frame* chegue no último nodo escravo, este o enviará de volta ao mestre, utilizando-se da comunicação *full-duplex* do Ethernet. Assim, o tempo de processamento de cada escravo é praticamente desconsiderável, fazendo com que essa solução apresente o determinismo desejado.

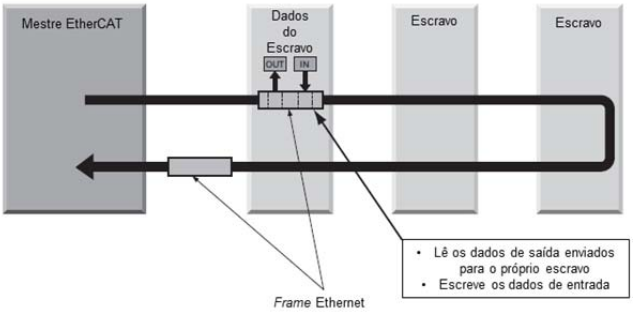


Figura 1. Comunicação EtherCAT (Fonte: Adaptado de [OMRON Automation Pvt Ltd 2016])

Embora o protocolo EtherCAT em si possua performance e determinismo ótimos, existem outros fatores externos ao protocolo que podem diminuir o desempenho. A meta deste trabalho é formular uma estratégia para determinar se a implementação atual da comunicação interfere na performance do equipamento.

3. Controladores Programáveis

CLPs são, em essência, computadores de propósito geral: recebem uma entrada, que é processada, e produzem uma saída, de acordo com sua programação. São otimizados para tarefas de controle, geralmente de tempo real, e para suportar ambientes industriais hostis. Isso inclui variações de temperatura, umidade, vibração mecânica, e ruído [Bolton 2015].

3.1. Entradas e Saídas

Um CLP precisa coletar informações do mundo externo para tomar decisões de controle. Essas informações chegam ao controlador através de suas entradas, que estão conectadas

a sensores. Similarmente, as decisões tomadas pelo sistema de controle chegam ao mundo exterior através dos módulos de saída, que são ligados a atuadores.

A decisão sobre usar I/O digital ou analógico depende da aplicação. Sinais discretos são adequadamente representados por entradas e saídas digitais, enquanto sinais contínuos são adequadamente representados por entradas e saídas analógicas. As entradas são lidas pelo CLP, que converte o valor lido para uma escala conveniente à aplicação.

A leitura de entradas, execução do programa e escrita de saídas são feitas de forma cíclica no tempo. O ciclo do CLP é apresentado na figura 2 [Bryan e Bryan 1997].

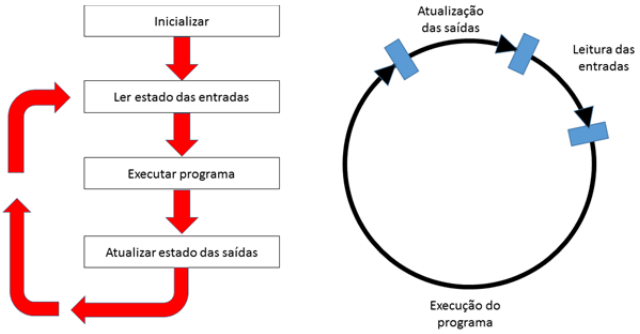


Figura 2. Ciclo do CLP

3.2. Execução: Tipos de Tarefas

Uma tarefa é um elemento de controle de execução capaz de invocar a execução de um conjunto de unidades de organização de programas (POUs), que incluem programas e blocos funcionais. A execução pode ser de forma periódica ou na ocorrência de eventos [IEC and others 1993]. Dessa forma, existem três tipos de tarefas:

- Tarefa Cíclica (*Cyclic Task*): executada em intervalos regulares de tempo. Este tipo de tarefa interromperá outras tarefas de prioridade mais baixa uma vez que tenha se esgotado o intervalo de tempo.
- Tarefa por Evento (*Event Task*): executada, apenas uma vez, na ocorrência da borda de subida da variável booleana especificada.
- Tarefa Contínua (*Continuous Task*): tarefa com execução contínua. O período de execução é definido pelo tempo de processamento gasto pela tarefa em um determinado momento. Portanto, o período de execução é variável, e será mais longo caso a tarefa precise de mais tempo de processamento.

4. Sistemas de Tempo Real e Determinismo

Um sistema de tempo real é definido como uma atividade ou sistema de processamento que precisa responder a entradas geradas externamente dentro de um período finito e especificado [Field-Richards 1983] (conforme citado em [Burns e Wellings 2001]). A dimensão deste tempo depende do tipo de aplicação.

No contexto de sistemas de tempo real, determinismo significa que o comportamento do sistema seja previsível em termos de valores e tempo [Kopetz 2011]. Em um sistema de frenagem, por exemplo, não é suficiente garantir que a ação de frenagem vai ser executada em algum momento posterior. Neste caso, seria necessário uma restrição mais forte, como definir que a ação ocorrerá 2 ms após o seu acionamento.

Assim, relacionando esses conceitos com os tipos de tarefas apresentados na seção 3.2, é possível perceber que tarefas contínuas não são adequadas para aplicações que necessitem de determinismo. Uma vez que não existe um tempo definido entre cada execução, não é possível saber antecipadamente em que momentos a tarefa executará.

4.1. Tempo de Ciclo

O tempo de ciclo é definido como o tempo entre a ativação da tarefa até o final da sua execução [Bolton 2015]. Isso inclui o tempo gasto pela atualização de I/O e o tempo de execução da lógica do usuário.

Os CLPs da série Nexto apresentam como tempo de intervalo mínimo entre ativações de 5 ms. Tanto aplicações como o sistema de frenagem mencionado na seção 4, quanto aplicações de *motion control*, precisam de tarefas que executem mais rapidamente.

Mesmo para uma tarefa que execute sempre o mesmo programa, existe uma variação neste tempo, chamada de *jitter* de tempo de ciclo. Atualmente, não é possível o uso desses CLPs nessas aplicações devido às características apresentadas pela série.

5. Trabalhos Relacionados

Sabendo-se que existem diversos componentes do sistema que podem ser otimizados em relação ao seu desempenho, o que se deseja é identificar qual deles é o gargalo. Neste trabalho foram usados equipamentos específicos. Por este motivo, não existem outros trabalhos especificamente relacionados com os mesmos dispositivos. Contudo, existem outros trabalhos de avaliação de desempenho que utilizam diferentes ambientes, ferramentas e equipamentos, ou mesmo trabalhos de cunho teórico.

Em [Jarp 2002], o autor propõe identificar gargalos em programas através do uso de *performance counters*. Com esse mecanismo de monitoração, são obtidas informações úteis sobre o programa em execução, como: quantidade de ciclos de processamento, número de instruções, ciclos de *stall* e atividade dos diversos níveis de cache. Essas informações são muito úteis para ajudar na identificação de gargalos.

Em [Jain 1990], são descritos critérios para a seleção de parâmetros para serem testados a fim de se identificar o gargalo do sistema. Parâmetros que não dizem respeito ao componente cujo desempenho se deseja melhorar podem ser omitidos, simplificando a análise. Também é descrito o uso de monitores, que são ferramentas utilizadas para observar atividades de uma determinada parte do sistema, coletando e analisando os dados.

6. Método Proposto

Sintetizando o que foi apresentado nas seções anteriores, o objetivo é tornar o tempo de ciclo mínimo menor, determinando se o desempenho poderia ser melhorado através de melhorias na implementação da comunicação do CLP com os módulos de I/O. Contudo, não podemos descartar a possibilidade de que a comunicação não seja o único ponto responsável pelo desempenho não satisfatório. O método proposto considera isso.

Primeiramente, deseja-se saber qual a ordem de grandeza do tempo de ciclo dada a implementação atual. Este é um passo inicial para avaliar se são necessárias otimizações e para estimar a magnitude das mudanças com base nas metas estabelecidas.

Caso seja detectada a necessidade de um melhor desempenho, pode-se prosseguir para identificar os gargalos. O método proposto é apresentado na figura 3 e tem o objetivo de servir como ferramenta para o processo de otimização. Os testes e as ações foram numerados na figura para auxiliar a compreensão.

Avaliar o desempenho sem comunicação (I): para determinar se o desempenho é de fato limitado pela comunicação, realiza-se um teste para avaliar o caso sem comunicação. Caso o desempenho não atinja a meta estabelecida, constata-se que a comunicação não é o principal gargalo. Sugere-se investigar funções de *callback* de outras funcionalidades do sistema que possam estar interrompendo a execução do programa. Além disso, sugere-se a utilização de um *profiler* para monitorar e analisar a execução. Se o desempenho for satisfatório, prosseguir para a avaliação (II).

Avaliar o desempenho com uma carga computacional constante (II): executar um programa de usuário sem comunicação com carga computacional constante, que ocupe uma parcela do tempo de execução da mesma ordem de grandeza do tempo ocupado pela comunicação. Observar se há *jitter* no tempo de ciclo. Com uma carga constante, idealmente o *jitter* seria pequeno. Caso o desempenho não seja aceitável, é possível que existam outras *threads* de usuário executando com prioridade maior causando interrupções no programa. Sugere-se investigar o escalonamento de tarefas, utilizando um *profiler* para determinar se a perda de desempenho é causada por problemas de prioridade. Se o desempenho for satisfatório, prosseguir para a avaliação (III).

Avaliar o desempenho com comunicação (III): se o desempenho for degradado apenas quando utilizado comunicação, significa que a comunicação é realmente um dos gargalos. Portanto, é importante tentar isolar qual das funcionalidades implementadas através de comunicação é o problema. Sugere-se uma abordagem que trabalhe com quatro diferentes partes da comunicação:

- Troca de I/O, responsável pela atualização de entradas e saídas e, portanto, de alta prioridade;
- Diagnósticos do CLP, de baixa prioridade;
- Comunicação relacionada ao *discovery cycle*, que é responsável pelo descobrimento de novos módulos;
- Interface de rede: possíveis problemas de gerenciamento de envio dos frames EtherCAT. Possível compartilhamento de recursos com a interface de rede que realiza a comunicação com o *software* de programação.

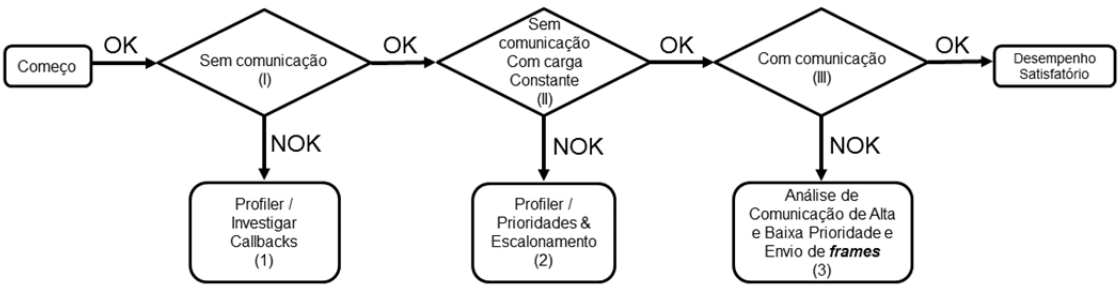


Figura 3. Fluxograma do método proposto

6.1. Resultados Preliminares

Conforme o método proposto, deseja-se avaliar a situação atual para determinar se mudanças são necessárias. Deseja-se fazer isso baseando-se em aplicações reais, utilizadas na prática, desses dispositivos. Assim, foi criada uma tarefa que não execute nenhum código de usuário, mas que realize comunicação. Deste modo, o tempo de ciclo da tarefa será composto apenas pelo tempo levado por outras funcionalidades, como descrito na seção 6, permitindo uma avaliação isolada da comunicação.

Como a meta do trabalho é possibilitar o uso da série Nexto em aplicações de alto desempenho, o intervalo de ativação foi definido como sendo o mínimo permitido pela ferramenta de programação, 5 ms, de modo a realizar a avaliação no contexto mais próximo dessas aplicações. As medidas de tempo de ciclo foram feitas segundo a definição apresentada na seção 4.1.

Foram realizados testes para determinar a quantidade de amostras que proporcionaria uma avaliação estável no que diz respeito à variabilidade dos tempos de ciclo. Assim, após variar o número de amostras, concluiu-se empiricamente que 150.000 amostras produziam um resultado estável, mostrado na tabela 2.

Tabela 2. Resultado do teste de tempo de ciclo.

Tempo de Ciclo Máximo (μs)	Tempo de Ciclo Mínimo (μs)	Tempo de Ciclo Médio (μs)
2686	1759	1918

Considerando que o tempo de ciclo máximo medido foi de cerca de 2.6 ms, é justificável que o intervalo de ativação mínimo da tarefa seja 5 ms. É preciso manter uma certa fatia de tempo para a execução do programa do usuário. Com esta configuração, no pior caso, tem-se aproximadamente metade do tempo de intervalo para execução de lógica. Tendo sido avaliada a situação atual e determinados os objetivos, o método proposto na seção 6 pode ser utilizado como ferramenta para identificar pontos a serem melhorados.

7. Conclusões

Neste artigo, foi feita uma revisão sobre assuntos relevantes ao estudo. Assim, foi consolidada a proposta, com a execução de testes e produção de resultados preliminares. A abordagem proposta neste trabalho tem como objetivo sistematizar uma metodologia que torne mais fácil a localização dos pontos a serem otimizados.

Conforme os resultados preliminares apresentados na seção 6.1, a série Nexto é passível de melhorias. Os testes realizados foram concebidos com o objetivo de observar o desempenho em uma situação similar às encontradas em aplicações reais. Através dos tempos de ciclo obtidos, com uso de parte da metodologia apresentada, foi constatada a necessidade de um melhor desempenho para o uso em determinados sistemas.

Os trabalhos futuros desenvolvidos a partir deste podem utilizar a metodologia para melhorar o desempenho, colocando-a em prática. Localizar a origem dos gargalos pode ser uma tarefa bastante complexa sem o uso de uma sistemática. A estratégia apresentada busca isolar diferentes componentes do sistema, tanto por serem de áreas diferentes, como comunicação e escalonamento, quanto isolar componentes dentro da mesma

área, como analisar a comunicação de alta e baixa prioridade. Assim, a análise é bastante simplificada. Apesar de o trabalho tratar de um estudo de caso, o método pode facilmente ser adaptado para outras implementações, tornando-o bastante versátil.

Referências

Beckhoff HW Documentation (2017). BECKHOFF CX9020 Hardware documentation. `https://infosys.beckhoff.com/content/1033/cx9020_hw/9007202790037387.html`. Acessado em 29/06/2017.

Beckhoff Product Overview (2017). BECKHOFF Fieldbus Components: Product Overview. `https://infosys.beckhoff.com/english.php?content=../content/1033/tcsample_crestron/html/tckb_crestron_performance.htm`. Acessado em 27/06/2017.

Beckmann, G. (2004). Ethercat communication specification, version 1.0. *EtherCAT technology group*.

Bolton, W. (2015). *Programmable logic controllers*. Newnes.

Bryan, L. A. e Bryan, E. A. (1997). *Programmable controllers: theory and implementation*. Industrial Text Company.

Burns, A. e Wellings, A. J. (2001). *Real-time systems and programming languages: Ada 95, real-time Java, and real-time POSIX*. Pearson Education.

Gupta, A. e Arora, S. (2009). *Industrial automation and robotics*. Laxmi Publications.

IEC and others (1993). Iec 61131-3. *Programmable Controllers-Part*, 3.

Jain, R. (1990). *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. John Wiley & Sons.

Jarp, S. (2002). A methodology for using the itanium 2 performance counters for bottleneck analysis. Technical report, Technical report, HP Labs.

Kopetz, H. (2011). *Real-time systems: design principles for distributed embedded applications*. Springer Science & Business Media.

Motion Controllers (2017). Motion controllers. `http://dl.mitsubishielectric.com/dl/fa/document/catalog/ssc/l03014/qmotion.pdf`. Acessado em 09/06/2017.

Neumann, P. (2007). Communication in industrial automationâĂŤwhat is going on? *Control Engineering Practice*, 15(11):1332–1347.

OMRON Automation Pvt Ltd (2016). Ethercat communication manual.

Série Nexto (2017). Série Nexto - Altus. `www.altus.com.br/ftp/Public/Portugues/Produtos/Nexto/00DocSerie/ManuaiseApostilas/MU214000.pdf`. Acessado em 29/06/2017.

Vosough, S. e Vosough, A. (2011). Plc and its applications. *International journal of multidisciplinary sciences and engineering*, 2(8).

PyCOO: Uma API em Python para Plataforma Click-On-OSv

Vinícius Fülber Garcia[1], Thales Nicolai Tavares[1], Leonardo da Cruz Marcuzzo[1], Lucas Bondan[2], Muriel Figueredo Franco[2], Giovanni Venâncio de Souza[3], Alberto Egon Schaeffer-Filho[2], Carlos Raniery Paula dos Santos[1]

[1]Departamento de Computação Aplicada – Universidade Federal de Santa Maria (UFSM)
Av. Roraima nº 1000 – 97.105-900 – Santa Maria – RS – Brazil

[2]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

[3]Departamento de Informática –Universidade Federal do Paraná (UFPR)
Caixa Postal 19.081 – 81.531-980 – Curitiba, PR – Brazill

{vfulber,tntavares,lmarcuzzo,csantos}@inf.ufsm.br,

{lbondan,mffranco,alberto}@inf.ufrgs.br, gvs11ufpr@gmail.com

Abstract. *The NFV technology is an option to keep a network function infrastructure in a more flexible, facilitated management and reduced costs way. Specialized platforms for VNFs execution, such Click-On-OSv, uses packet accelerators to recover part of the lost performance. This work presents PyCOO, an API developed in Python that enables the control from infrastructure to Click-On-OSv's internal routines to facilitate remote utilization and prototyping environments creation.*

Resumo. *A tecnologia NFV é uma opção para manter uma infraestrutura de funções de rede de forma mais flexível, com gerência facilitada e custos reduzidos. Plataformas especializadas na execução de VNFs utilizam aceleradores de pacotes para recuperar parte desse desempenho perdido, como é o caso do Click-On-OSv. Este trabalho apresenta o PyCOO, uma API desenvolvida em Python que possibilita o controle desde da infraestrutura até as rotinas internas do Click-On-OSv a fim de facilitar a utilização remota da mesma e criação de ambientes de prototipação.*

1. Introdução

Funções Virtualizadas de Rede (NFV) se tornaram um paradigma que objetiva desacoplar as funções de rede de seu hardware dedicado. Para que isso seja possível sistemas de virtualização já existentes são utilizados (*i.e., hypervisor*) para instanciar servidores virtuais, estes servidores rodam em *hardware* genérico [ETSI 2012], o que contribuí para reduzir o *time to market* e custos de capital e operacionais (CAPEX e OPEX) [Mechtri et al. 2017].

Um conjunto ordenado de funções virtualizadas de rede e elementos que definem rotas pelas quais pacotes e fluxos definem uma Cadeia de Funções de Serviço (SFC) [Quinn and Nadeau 2015]. Do ponto de vista operacional uma SFC pode ser descrita através de um grafo chamado de Grafo de Encaminhamento (VNF-FG)[ETSI 2013], este

grafo contém nós representando entidades como VNFs e arestas como conexões lógicas. Pontos de entrada e saída de dados também são definidos em um VNF-FG e servem para indicar o inicio e o fim da SFC.

Apesar do paradigma NFV apresentar diversas características vantajosas em relação a utilização de *middleboxes*, como facilidade para realização de operações de escala e maior liberdade de desenvolvimento, o uso de *hardware* genérico para a execução de funções de rede apresenta desempenho inferior quando comparada ao uso de *hardware* dedicado [Hwang et al. 2015]. Esforços para reduzir essa diferença de desempenho são realizados através do desenvolvimento de plataformas de software específicas para hospedar e executar funções de rede virtualizadas.

Plataformas para execução de NFV como ClickOS [Martins et al. 2014], OpenNetVM [Zhang et al. 2016] e Click-On-OSv [Marcuzzo et al. 2017], utilizam tecnologias de aceleração de pacotes Netmap [Rizzo 2012] ou Intel DPDK [Intel 2017]. O desenvolvimento e otimizações das plataformas é uma etapa fundamental para a disseminação e popularização do paradigma.

Diversas soluções para controle de infraestrutura estão disponíveis [Baton 2017][Sefraoui et al. 2012][Tacker 2017]. Essas aplicações têm capacidade de realizar a instanciação e manter a infraestrutura de máquinas virtuais que hospedam funções de rede virtualizadas. Entretanto, essas ferramentas não são capazes acessar as plataformas internamente para manipularem as funções a serem executadas, para isso sistemas auxiliares que se adaptam as peculiaridades de cada plataforma são desenvolvidos.

Este trabalho apresenta uma API desenvolvida em Python para acesso e controle de VNFs que executam sobre a plataforma Click-On-OSv. A API consiste de dois grandes módulos (infraestrutura e funções virtualizadas). O módulo de infraestrutura tem como responsabilidade o gerenciamento de Máquinas Virtuais (VMs) enquanto que o módulo de funções virtualizadas é responsável pelo gerenciamento de VNFs, SFCs e pela aplicação de *Scripts* de Execução.

A API elaborada pode ser utilizada para diversos tipos de aplicações que necessitem instanciar e controlar VNFs. A partir do conjunto de funções dispostas, é possível criar ambientes de prototipação da plataforma Click-On-OSv semelhantes ao ESCAPE [Csoma et al. 2014]. Também, a solução pode ser utilizada para a criação de topologias de rede de maneira automática para a realização de testes, como comparações de desempenho de diferentes implementações de uma mesma função em Click.

O restante desse artigo está organizado como se segue: a seção 2 expões as características de acesso e controle das principais ferramentas e plataformas para execução de NFV. A seção 3 trás em detalhes os requisitos e o modo de funcionamento da API desenvolvida. A seção 4 apresenta a conclusão do trabalho.

2. Plataformas e Interfaces de Acesso e Controle

Diferentes plataformas para a execução de VNFs estão disponíveis. Essas plataformas apresentam diferentes meios para gerenciamento das funções as quais se destinam a executar. A simplificação das interfaces de gerência das funções virtualizadas é um passo fundamental para a popularização da tecnologia NFV.

O Click Modular Router [Kohler et al. 2000] utiliza uma arquitetura flexível e modular de software para criar e encaminhar pacotes. Sua arquitetura baseia-se na utilização de elementos específicos para processamento de pacotes, com funcionalidades específicas (*e.g.,* entrada de pacotes, classificadores L2 e L3). Estes elementos podem ser interconectados para a criação de novas funcionalidades complexas (*e.g., Switches*, Roteadores).

Cada elemento implementa uma função simples, a qual possui um ponto de entrada e saída e pode ser configurado individualmente, através de *handlers* internos. O gerenciamento de uma função de rede executando no Click é feita através de um elemento chamado `ControlSocket`. Este elemento expõe, através de um socket TCP, os *handlers* de cada um dos elementos que compõem a função, que por sua vez podem ser configurados individualmente através de um agente externo. Alguns dos *handlers*, como contadores, podem ser utilizados para coleta de métricas de desempenho da função.

O ClickOS [Martins et al. 2014] é uma plataforma desenvolvida para a execução de VNFs por meio de de virtualização utilizando o Mini-OS com drivers paravirtualizados modificados para suportar o *netmap*, este último é uma solução para realização de entrada e saída de pacotes em alta velocidade. Em relação a criação e execução das VNFs, a solução Click Modular Router é utilizada, este executa as funções de rede. Sua arquitetura têm como foco o flexibilidade, desempenho e escalabilidade.

A gerência de uma instância do ClickOS é realizada através da modificação de variáveis internas utilizando o XenStore. Esta interface permite apenas o controle do ciclo de vida da instância e o *upload* de funções, de forma que os *handlers* internos do Click não podem ser configurados (*i.e.,* não há um `ControlSocket` em execução). Assim, não é possível coletar métricas internas, e alterações em elementos implica no *upload* de uma nova função de rede.

A plataforma OpenNetVM [Zhang et al. 2016] emprega o uso de *containers* Docker para a execução de VNFs em servidores em conjunto com o *framework* de aceleração de pacotes DPDK, o qual objetiva minimizar a sobrecarga do processamento de pacotes e fornecer redes com grandes taxas de transferência e baixa latência em ambientes virtualizados [Ramakrishnan 2016]. A arquitetura desta plataforma é composta de dois módulos principais: *NF Manager* e as NFs. O *NF Manager* é responsável por manter o estado da rede através da gerência de NFs e roteamento dos pacotes. Já as NFs se comunicam com o *NF Manager* através da *NFLib*, informando quando estão prontas para serem executadas.

Nota-se que as NFs do OpenNetVM não são implementadas utilizando o Click, e devem sem implementadas em linguagem C, utilizando *callbacks* da *NFLib* para definir pontos de entrada, processamento e saída. Assim, não existem elementos prontos que podem ser utilizados para a construção de uma NF, bem como não existe uma forma pré-definida de recuperar métricas e reconfigurar NFs pois esses devem ser definidos durante a implementação das mesmas.

A plataforma Click-On-OSv (COO) utiliza o sistema operacional minimalista OSv [OSv 2017] executando o Click Modular Router[Kohler et al. 2000] e o acelerador de pacotes Intel DPDK para desempenhar funções virtualizadas de rede. O Click-On-OSv se diferencia de seus similares por prover tanto uma interface gráfica quanto uma interface REST de forma nativa para gerenciamento de VNFs.

Para a gerência do Click-On-OSv, é possível utilizar tanto requisições REST como

uma interface Web. A interface REST é implementada como uma extensão da interface nativa do OSv, e permite controle do ciclo de vida, tanto da função virtualizada como da instância da máquina virtual. Internamente, um `ControlSocket` conecta a instância do Click com a interface REST do OSv, o que permite que métricas internas das funções possam ser coletadas externamente através da interface do OSv. A interface Web, por sua vez, é uma interface gráfica para os métodos REST, o que permite que as métricas coletadas possam ser exibidas através de gráfico, com botões para controle do ciclo de vida, propriedades dos descritores das VNFs. A plataforma também implementa um editor de funções e um *log* de erros, que podem ser utilizados para o tratamento de eventuais problemas que ocorram na instanciação da função.

3. PyCOO

A API para acesso e controle de VNFs foi desenvolvida com o intuito de facilitar o gerenciamento da plataforma Click-On-OSv e facilitar a realização de tarefas através da definição de comandos de alto nível compostos por um conjunto de comandos básicos. Além disso, por se tratar de um grupo de funções que podem se importadas, a API como um todo ou apenas módulos dela podem ser utilizados como parte da implementação de softwares terceiros.

A API foi desenvolvida em Python, utiliza o *hypervisor* KVM para oferecer a infraestrutura de instanciação necessária, Linux *Bridges* para prover conexões entre VNFs para a formação da SFC e também com elementos externos a mesma, além de fazer uso de chamadas REST para executar ações internas a plataforma. Toda a comunicação entre usuário e API pode ser realizada através arquivos de configuração em formato JSON.

A Figura 1 apresenta a estrutura da API, os elementos em azul representam os módulos implementados como parte deste trabalho, em vermelho são apresentados agentes previamente existentes necessários para a utilização da API, em verde são apresentados os arquivos de configuração que definem configurações das VNFs, SFCs e as ações a disponibilizadas que são consumidos pela API.

A API [1] é dividida em quatro módulos, de VMs, de VNFs, de SFCs e de *Scripts* de Execução. Cada módulo é acoplado e utiliza dados providos por um terceiro, a exceção do controle de VMs que é independente e substituível. Uma instância virtual executando o sistema Click-On-OSv é caracterizada como uma VNF, um conjunto de VNFs e informações de rotas entre elas designa uma SFC, por fim, *scripts* de execução indicam sequências de ações a serem tomadas, também podem ser definidas situações de exceção.

3.1. Gerenciamento de VMs

A API utiliza uma imagem padrão da plataforma Click-On-OSv, todas as VNFs apresentam um descritor simples que altera o arquivo de instanciação usado pelo KVM através da CLI Virsh, os dados contemplados pelo descritor são:

- ID: identificador único, será o nome dado ao diretório contendo os arquivos da VNF e imagem da VM;
- *Memory*: quantidade dedicada de memória RAM para a VM que será instanciada;
- VCPU: número de CPUs virtuais dedicadas a execução da VM;

[1] Código e exemplos: https://github.com/ViniGarcia/ClickOnOSvManager

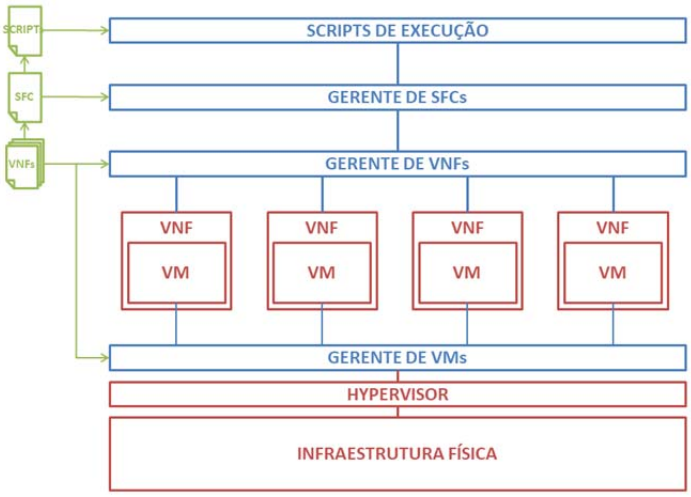


Figura 1. Elementos e Relações Referentes a API

- *Management MAC*: Endereço MAC dedicado para a interface de gerência da plataforma;
- *Interfaces*: cada interface contém um ID que indica a Linux *Bridge* a ser utilizada e um MAC dedicado a ela, essa informação é usada somente quando a instanciação for de uma VNF isolada e não de uma SFC.

As VMs são criadas em modo temporário quando iniciadas, ou seja, seu ciclo de vida começa quando são instanciadas e termina quando são desligadas. Após a configuração da VM ser lida, um objeto do tipo VNF é criado, sendo disponibilizo métodos de controle da VM, como ligar e desligar.

Métodos para mudança de configuração de VNF e obtenção de informações da VM também são disponibilizados. Finalmente, métodos para a criação do diretório da VNF e cópia de seus subsequentes arquivos e para a destruição dos mesmos são oferecidos para possibilitar o reuso de IDs e economia de memória.

3.2. Gerenciamento de VNFs

Consiste no gerenciamento das funções que a plataforma Click-On-OSv é capaz de realizar, é feito através da interface REST disponibilizada pela plataforma acessada via requisições HTTP enviadas através da API. Entre as ações que podem ser realizadas estão:

- *POST*
  - *Start*: ativa o DPDK e o Click, após executa a função que está reservada dentro da plataforma;
  - *Stop*: para a função Click que esta executando na plataforma;
  - *Function*: ao enviar o caminho do arquivo que contém uma função Click, a mesma é repassada a plataforma e passa a ser a função reservada.
- GET
  - *Running*: retorna informação do estado do módulo Click, se o mesmo esta ativo ou não;

– *Function*: retorna a função de rede reservada nas configurações internas da plataforma;
– *Identification*: retorna os metadados contidos na especificação da função de rede reservada;
– *Metrics*: retorna métricas sobre a execução da função de rede na plataforma;
– *Log*: retorna informações de *log* sobre inicialização, execução e conclusão de uma função de rede.

3.3. Gerenciamento de SFCs

O gerenciador de SFCs utiliza um descritor que apresenta dados básicos de VNFs a serem utilizadas e conexões entre as mesmas. Antes de ser submetida, todos os dados presentes no documento de descrição são submetidos a uma análise de integridade quanto a existência das VNFs requisitadas e do grafo gerado. As informações necessárias para a definição de uma SFC são:

- ID: identificador único, será o nome pelo qual o programa identificará a SFC quando em execução;
- VNFs: conjunto identificando todas as VNFs que fazem parte da SFC. É necessário que o arquivo de configuração das VNFs escolhidas esteja presente no sistema;
- *Incoming Point* (IP): ponto de entrada de dados. Existe apenas um para cada SFC, recebe dados de uma fonte externa a SFC e repassa a primeira VNF do sistema, é definido por um ID e uma ponte;
- *Outcoming Points* (OPs): pontos de saída de dados, ou seja, pontos finais onde dados são transmitidos de uma VNF para um destino externo a SFC. Como o IP, é definido por um ID e uma ponte;
- *Connections*: cada conexão é definida por um *Input Logical Link* (ILL) ou seja, uma fonte de dados, e um *Output Logical Link* (OLL) que representa um destino para os dados pós processamento. A exceção de quando o ILL é o IP e quando o OLL é um OP, um nome para a ponte e os MACs das interfaces que receberão se conectarão a mesma devem ser definidos.

A validação de grafo utiliza regras para garantir que existe um grafo totalmente conectado e sem possíveis *loopings* infinitos, que o IP contenha apenas uma conexão com uma VNF afim de enviar dados (OLL) e nunca sejam utilizados como entrada de dados (ILL), que os OPs sejam utilizados como entrada de dados por VNFs (ILL) e nunca sejam utilizados como saída de dados (OLL), além de garantir que todos os elementos previamente definidos figurem no grafo da SFC.

São dispostos métodos para controle da SFC como um todo com ações padrão como ligar, desligar, criar, modificar e destruir, além de possibilitar que o gerente da cadeia de funções de rede virtualizadas acesse individualmente cada VNF afim de utilizar sua respectiva interface REST através da API e permitir a submissão de *scripts* de execução a serem aplicados a SFC como um todo.

3.4. *Scripts* de Execução

Um *script* de execução representa ações de alto nível descritas através de um conjunto de chamadas REST realizadas de forma serial destinadas a uma VNF específica.A execução

do *script* é realizada atravésda classe SFC com a identidicação da ação a ser excutada. A instância da SFC pesquisa todas as VNFs presentes no *script* que estão ativas e que contenham a ação requisitada e repassa os comandos a serem executados. A definição de uma ação de exceção em caso de erro é opcional.

A definição de um *script* de execução é dada por:

- *ID*: como um *script* de execução pode determinar ações diferentes para VNFs diferentes, o ID da VNF deve ser explícito determinando a quem se destina as definições;
- *Function*: função de rede virtualizada escrita em Click que será levada em consideração para a realização das ações;
- *Path*: local onde o arquivo da função nomeada no item anterior está localizada;
- *Actions*: consiste de um identificador da ação que pode ser requisitada por um agente externo, cada ação consiste de um conjunto de chamadas REST adicionada de sua precedência.

Como alternativa, um dicionário pode ser passado a classe VNF contendo apenas as chamadas REST e sua precedência para serem executadas de maneira serial, nesse caso, dispensa-se a definição de um arquivo JSON para o script e realiza-se as requisições puramente via código.

4. Conclusão

Com a crescente popularização de NFV, também cresce a demanda por *enablers* capazes de manter e executar uma infraestrutura baseada nesta tecnologia. Click-On-OSv é uma plataforma capaz de gerenciar e executar funções de rede virtualizadas implementadas em Click, sendo que eu diferencial reside na disponibilização de interfaces Web e REST nativa de gerencia.

Este trabalho apresenta uma API em Python que objetiva facilitar o gerenciamento tanto de uma VNF específica quanto da SFC como um todo. A API conta com quatro módulos que se complementam, são eles o módulo de VM, de VNF, de SFC e de *Scripts* de Execução.

Em trabalhos futuros objetiva-se aprimorar a API para trabalhar com VNFs compartilhadas, permitindo a utilização da mesma VNF entre duas ou mais SFCs. A flexibilização das tecnologias para instanciação de VMs e conexão entre as mesmas, permitindo a utilização de diferentes *hypervisors* e de Open vSwitch, também é finalidade deste projeto.

Ainda como trabalhos futuros, alterações no módulo de execução de *scripts* serão projetados a fim de permitir a definição de uma exceção global, dessa forma ações como de *rollback* do sistema podem ser consideradas. Por fim, objetiva-se flexibilizar a definição das sequências de ações através da hierarquização de processos considerando um cenário de sucesso ou falha para cada instrução.

Referências

Baton, O. (2017). Open baton: An open source reference implementation of the etsi mano. `https://openbaton.github.io`. Accessed: 2017-07-11.

Csoma, A., Sonkoly, B., Csikor, L., Németh, F., Gulyas, A., Tavernier, W., and Sahhaf, S. (2014). Escape: Extensible service chain prototyping environment using mininet, click, netconf and pox. In *ACM SIGCOMM Computer Communication Review*, volume 44, pages 125–126. ACM.

ETSI, G. (2013). Network functions virtualisation (nfv); use cases. *V1*, 1:2013–10.

ETSI, N. (2012). Network functions virtualization, white paper.

Hwang, J., Ramakrishnan, K. K., and Wood, T. (2015). Netvm: High performance and flexible networking using virtualization on commodity platforms. *IEEE Transactions on Network and Service Management*, 12(1):34–47.

Intel (2017). Intel data plane development kit. `http://http://dpdk.org`. Accessed: 2017-05-27.

Kohler, E., Morris, R., Chen, B., Jannotti, J., and Kaashoek, M. F. (2000). The click modular router. *ACM Transactions on Computer Systems (TOCS)*, 18(3):263–297.

Marcuzzo, L. d. C., Garcia, V. F., Cunha, V., Corujo, D., Barraca, J. P., Aguiar, R. L., Schaeffer-Filho, A. E., Granville, L. Z., and Santos, C. R. P. d. (2017). Click-on-osv: A platform for running click-based middleboxes. In *2017 IFIP/IEEE International Symposium on Integrated Network Management*. IEEE.

Martins, J., Ahmed, M., Raiciu, C., Olteanu, V., Honda, M., Bifulco, R., and Huici, F. (2014). Clickos and the art of network function virtualization. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, pages 459–473, Seattle, WA. USENIX Association.

Mechtri, M., Ghribi, C., Soualah, O., and Zeghlache, D. (2017). Nfv orchestration framework addressing sfc challenges. *IEEE Communications Magazine*, 55(6):16–23.

OSv (2017). Osv - the operating system. `http://osv.io`. Accessed: 2017-07-11.

Quinn, P. and Nadeau, T. (2015). Problem statement for service function chaining.

Ramakrishnan, K. (2016). Software-based networks: Leveraging high-performance nfv platforms to meet future communication challenges. In *Distributed Computing Systems (ICDCS), 2016 IEEE 36th International Conference on*, pages 24–24. IEEE.

Rizzo, L. (2012). Netmap: a novel framework for fast packet i/o. In *21st USENIX Security Symposium (USENIX Security 12)*, pages 101–112.

Sefraoui, O., Aissaoui, M., and Eleuldj, M. (2012). Openstack: toward an open-source solution for cloud computing. *International Journal of Computer Applications*, 55(3).

Tacker (2017). Tacker: Openstack nfv orchestration. `https://wiki.openstack.org/wiki/Tacker`. Accessed: 2017-07-12.

Zhang, W., Liu, G., Zhang, W., Shah, N., Lopreiato, P., Todeschi, G., Ramakrishnan, K., and Wood, T. (2016). Opennetvm: A platform for high performance network service chains. In *Proceedings of the 2016 Workshop on Hot Topics in Middleboxes and Network Function Virtualization*, HotMIddlebox '16, pages 26–31, New York, NY, USA. ACM.

Análise de Desempenho do KVM Aplicado a Paravirtualização e Virtualização Assistida por Hardware Embasada pela RFC 2544

Everson Luis R. Lucion[1], Giulliano G. Minuzzi[1], Mauricio V. Almeida[1]

[1]Centro de Processamento de Dados (CPD)
Universidade Federal de Santa Maria (UFSM) – Santa Maria, RS – Brasil

{everson,gminuzzi,mauricio}@cpd.ufsm.br

Abstract. *Virtualization is essential for consolidating datacenters, NFV and cloud computing. In this work, experiments with KVM and Intel VT technology resources were performed to identify the best virtualization technique regarding paravirtualization and hardware-assisted virtualization based on RFC 2544. Throughput, loss and latency metrics were evaluated. The results show that the gains apply only to the features of the Processor. Network throughput in emulated devices does not show significant performance gain due to limitation of vCPU usage by Qemu and poor performance in binary translation. Paravirtualization has better performance, but the virtualized machine needs to be modified to be compatible, which can lead to compatibility and portability issues.*

Resumo. *A virtualização é essencial para consolidação datacenters, NFV e computação em nuvem. Neste trabalho foram realizados experimentos com o KVM e recursos da tecnologia Intel VT para identificar a melhor técnica de virtualização no que tange a paravirtualização e virtualização assistida por hardware, embasados pela RFC 2544. As métricas de throughput, perdas e latência foram avaliadas. Os resultados mostram que os ganhos se aplicam somente aos recursos do Processador. O throughput de rede em dispositivos emulados não apresenta ganho de desempenho significativo devido a limitação de uso de vCPU pelo Qemu e desempenho fraco na tradução binária. A paravirtualização apresenta melhor desempenho, porém a máquina virtualizada precisa ser modificada para ser compatível, podendo ocasionar problemas de compatibilidade e portabilidade.*

1. Introdução

A virtualização é a tecnologia fundamental em centros de dados e computação em nuvem. Empresas e instituições acadêmicas buscam consolidar seus datacenters e reduzir custos. Soluções como NFV (Network Functions Virtualization) também chegaram para impulsionar a virtualização de serviços como NAT, *firewall*, IPS, DNS e *caching*.

O Virtual Machine Monitor (VMM) é o componente principal da virtualização, a sua eficácia afeta o desempenho de todo sistema diretamente, podendo executar na maioria das vezes a paravirtualização ou virtualização assistida por *hardware*. O *Kernel-based Virtual Machine* (KVM) é uma solução completa de virtualização em

código aberto que suporta extensões de virtualização (Intel VT ou AMD-V). O KVM foi integrado ao Kernel do Linux em 2007 [Carvalho and Bellezi 2014].

As equipes de TI sempre estão preocupadas com o viés do desempenho. A busca pela otimização e ganho é constante, suscitando dúvidas em relação a melhor abordagem de virtualização a ser utilizada.

Neste trabalho foram realizados experimentos com máquinas virtualizadas pelo KVM e recursos da tecnologia Intel VT visando identificar a melhor técnica de virtualização, obtendo assim melhor desempenho nas operações de *network* I/O em enlaces *Gigabit.* Os experimentos abordam as métricas de vazão, perdas e latência embasados pela RFC 2544 que recomenda tamanhos de *frames* variados (64, 128, 512, 1024, 1208 e 1518 bytes). Todos os tamanhos de *frames* são usados em uma rede e os resultados devem ser conhecidos. A comparação é realizada entre o *host* físico e duas máquinas virtualizadas, uma paravirtualizada (driver Virtio) e outra com virtualização assistida por hardware (driver emulado pelo Qemu). A máquina física foi incluída no experimento servindo com parâmetro de desempenho ideal.

Os resultados mostram que apesar da Tecnologia Intel VT ter sido desenvolvida para "otimizar" o trabalho do VMM, os ganhos de performance se aplicam somente ao uso dos recursos do Processador. Os recursos de *throughput* de rede em dispositivos emulados não apresentam ganhos de performance significativos devido a limitação de uso de vCPU pelo Qemu e fraco desempenho da tradução binária. A paravirtualização apresenta melhor desempenho, porém a máquina virtualizada precisa ser modificada para ser compatível, podendo ocasionar problemas de compatibilidade e portabilidade.

Este trabalho está estruturado da seguinte forma: A Seção 2 discute os trabalhos já relacionados, a Seção 3 expõe o método, a técnica e as tecnologias que integram o experimento. A Seção 4 apresenta a metodologia proposta e a Seção 5 mostra os resultados dos experimentos. A Seção 6 conclui e discute trabalhos futuros.

2. Trabalhos Relacionados

Motika and Weiss (2012) analisaram através da ferramenta Iperf o desempenho do KVM com dispositivos de redes emulados e paravirtualizados. Relataram que a melhoria do *driver V*irtio na largura de banda de transmissão é de 54% e na largura de banda recebida é 58% em relação ao *driver* emulado. As máquinas virtuais apresentaram um melhor desempenho na recepção de dados e que a transmissão não teve desempenho satisfatório. A utilização da CPU foi alta em ambos ambientes, mais de 90% de utilização de CPU no *driver* paravirtualizado e um pouco maior no *driver* emulado. Não informaram largura de banda e não fizeram uso da RFC 2544.

Mais próximo a este trabalho está a proposta de Kolling et al. (2008), o qual teve como principal objetivo implementar os experimentos da RFC 2544 em sistema embarcado, que é um conjunto computacional dedicado e usa seu poder de processamento exclusivo para a função a qual foi projetado. Comparou o desempenho em testadores implementados em *software* com o protótipo desenvolvido em *hardware,* com o objetivo de validar a vazão máxima alcançada em cada um. Relatou que o experimento em *hardware* não alcançou a vazão máxima apenas para *frames* de 64 *bytes*, atingindo 71% do ideal. Em relação aos experimentos por *software*, todos os tamanhos de *frames* tiveram desempenho inferior, e que esta diferença se deve ao

elevado uso de CPU pelo aplicativo no processamento dos cabeçalhos dos *frames* Ethernet.

A nossa proposta apresenta em relação às anteriores, três inovações. Primeiro, analisa o desempenho e aspectos positivos e negativos de cada técnica de virtualização. Em segundo lugar identifica qual técnica apresenta melhor desempenho. Por último, relata porque a conversão binária fraca é o principal problema da virtualização.

3. A Virtualização por Método VMM

O KVM utiliza a tecnologia Intel VT em todas as técnicas de virtualização para aumento de performance nas operações ligadas ao processador, executando diretamente as operações das VMs no mesmo, evitando a tradução binária. Por outro lado o gerenciamento de dispositivos é efetuado pelo Qemu quando utilizada a virtualização total, já na paravirtualização os mesmos são gerenciados pelos Virtio *drivers.* O VMM é também conhecido como *hypervisor* e neste experimento muitas vezes é referenciado como máquina física.

3.1. Tecnologia de Virtualização Intel VT

Desenvolvida para oferecer melhor performance, a virtualização assistida por *hardware (HVM)* requer extensões de *hardware* Intel VT ou AMD-V, possibilitando que o VMM execute de forma isolada máquinas virtuais (VMs) com qualquer sistema operacional, necessitando que o processador ofereça suporte a tecnologia [Patel and Chaudhary 2014]. De acordo com Motika and Weiss (2012), a introdução desta tecnologia nos processadores visa reduzir o *overhead* gerado pela troca de contexto entre o VMM e o sistema hospedeiro, através da utilização de uma estrutura de dados que retém informações sobre o estado da CPU (*Virtual-Machine Control Structure* - VMCS).

A virtualização assistida por *hardware* substitui a tradução binária por dois modos de operação no processador, sendo estes o *root* que opera de forma semelhante ao processador habitual, responsável pela execução do VMM, e o *non-root* responsável somente pela execução de atividades das VMs, os quais, também, oferecem suporte aos quatro *rings* da arquitetura x86-64, porém o modo *non-root* dispõe de menos privilégios em qualquer *level* dos *rings* [Grinberg and Weiss 2012].

3.2. Técnicas de Virtualização

A Paravirtualização é uma técnica que consiste em modificar o núcleo do sistema convidado (*guest*) substituindo as chamadas privilegiadas do SO por chamadas ao *hypervisor* (*hypercalls*) [Shirinbab, Lundberg e Ilie 2014]. Dessa forma, o *guest* tem acesso direto ao *hardware* através da utilização de *drivers* que se comunicam com o *back end driver* do VMM. Na paravitualização, a VM precisa ser modificada para ser compatível, podendo ocasionar problemas de compatibilidade e portabilidade. No KVM, a paravirtualizacão é baseada na utilização dos Virtio *drivers,* separando o *driver* do *guest* da implementação por parte do *hypervisor* [Zhang et al. 2010].

Virtualização completa, também conhecida como emulada, executa um g*uest* sem necessidade de modificações de seu *kernel*, recebendo dispositivos emulados pelo VMM, o que garante a portabilidade e compatibilidade com outros sistemas.

A Intel agregou a virtualização assistida por *hardware* a seus processadores através das tecnologias Intel VT, para que o *hypervisor* possa mais eficientemente delegar o acesso a recursos restritos. A Intel VT otimiza a execução de operações gerenciadas pelo processador, mas seus *drivers de* rede (placas *ethernet*) e bloco (*drivers* de disco), entre outros, ainda são emulados pelo VMM e gerenciados pelo Qemu. *Drivers* como o e1000 são atribuído ao *guest* pelo VMM neste experimento. Existem outras opções como vmxnet3 e rtl8139.

4. Metodologia da Avaliação de Desempenho

O protocolo Ethernet IEEE 802.3 é usado em redes de computadores e telecomunicações, define a camada física e enlace. Neste sentido, conexões *ethernet* podem ser testadas para garantir o nível de desempenho necessário. A avaliação de desempenho e a forma dos resultados, podem ser embasados em metodologias definidas pela IETF e publicadas através da RFC 2544 (COSTA, 2008).

A RFC 2544 é um padrão internacional formulado pela RFC (Request For Comments) para avaliar o desempenho de dispositivos de redes. A RFC 2544 orienta que utilizem-se *frames* nos tamanhos 64, 128, 256, 512, 1024, 1280 e 1518 *bytes*, enviados durante um intervalo de tempo e por um número definido de repetições. Os *frames* menores têm uma vazão efetiva menor do que o dos *frames* maiores, devido à inclusão dos *bytes* de cabeçalho e espaço entre *frames*, não sendo contabilizados como dados.

Para implementar o conjunto dos experimentos, a RFC 2544 define um ***tester*** (dispositivo testador) um **DUT** (dispositivo sob teste) do inglês "*device under test*", conforme pode ser visto na Figura 1.

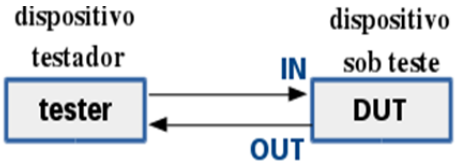


Figura 1. *tester e* DUT

4.1. Métricas de Avaliação de Desempenho

Muitas vezes pode ser essencial a realização de experimentos de desempenho, onde é necessário analisarmos determinados cenários de operações e observarmos os resultados obtidos. As métricas de desempenho avaliadas neste experimento são as seguintes:

- *Throughput* ou vazão: Taxa mais rápida na qual a contagem de *frames* reflete o máximo de tráfego de dados que podem ser manipulados sem perdas;
- Latência: A medida do atraso da extremidade de uma rede (*link*) ou dispositivo e vice-versa. O atraso é a soma de processamento nos elementos de rede e dos atrasos de propagação ao longo do meio de transmissão;
- Taxa de perda: A taxa de perda refere-se à porcentagem de *frames* que nunca foram recebidos em relação aos *frames* que foram transmitidos com sucesso a partir da fonte;
- Desempenho da CPU: Foram recolhidos dados sobre o desempenho das CPUs

durante os experimentos de t*hroughput,* nos eventos de transmissão e recepção do DUT.

4.2. Descrição dos Experimentos

Neste trabalho, a aferição do *throughput* ou vazão usou a ferramenta de *benchmarking* Nuttcp e o protocolo UDP com tamanhos de *frames* definidos pela RFC 2544 em cada direção do DUT (transmissão e recepção). Foram efetuados 10 experimentos consecutivos com 60 segundos calculando-se a sua média em Mbps.

Nos experimentos de latência, utilizou-se a ferramenta "ping" para efetuar 10 sequencias com envio de 100 *frames* conforme a RFC 2544. Para medir a latência, os experimentos foram feitos através de um *loop* e apresentados em milisegundos.

Na taxa de perda, o DUT foi aferido conforme a RFC 2544. Neste experimento também foi utilizada a ferramenta Nuttcp. Foram efetuados 10 eventos consecutivos com 60 segundos cada e a média foi apresentada em porcentagem.

O Nuttcp é uma ferramenta de medição de desempenho de rede. O uso mais comum é determinar o *throughput* da camada de rede TCP (ou UDP). Ele fornece informações adicionais, como a utilização da CPU do transmissor e do receptor

Os experimentos foram realizados em três cenários. O primeiro é o ***tester***: SGI 1U, RAM de 13 GB, 8 @ 2.00GHz. O segundo é o **VMM:** DELL PowerEdge R620, RAM 129 GB, 32 @ 2.60GHz. Em terceiro, as duas **VMs**, com 4 processadores, memória RAM de 8 GB cada e com interfaces Gigabit Ethernet. O KVM-QEMU versão 2.1.2, Nuttcp versão 6.1.2. e o SO Debian Jessie 3.16.0-4-amd64.

5. Resultados e Discussão

Na Figura 2(a), temos os resultados relacionados a taxa de vazão (DUT IN). Nota-se que a vazão da máquina física ou (VMM) e a paravirtualizada (PV) se mantiveram relativamente iguais. A virtualização assistida por *hardware* (HVM) apresentou um desempenho excelente nos *frames* de 1280 bytes (949.11 Mbps) e 1518 *bytes* (952.33 Mbps), e satisfatório nos demais tamanhos.

O desempenho da HVM não foi considerado satisfatório no (DUT OUT) em todos tamanhos de *frames* testados. Este comportamento pode ser explicado por conta do *driver* emulado, em que todo o I/O é manipulado pelo Qemu, pela limitação de uso de vCPU e fraco desempenho na tradução binária. A HVM apresentou uma vazão máxima de 189.94 Mbps com *frames* de 1518 *bytes*. A VMM e a PV apresentaram excelentes resultados. A Figura 2(b) demonstra estes resultados.

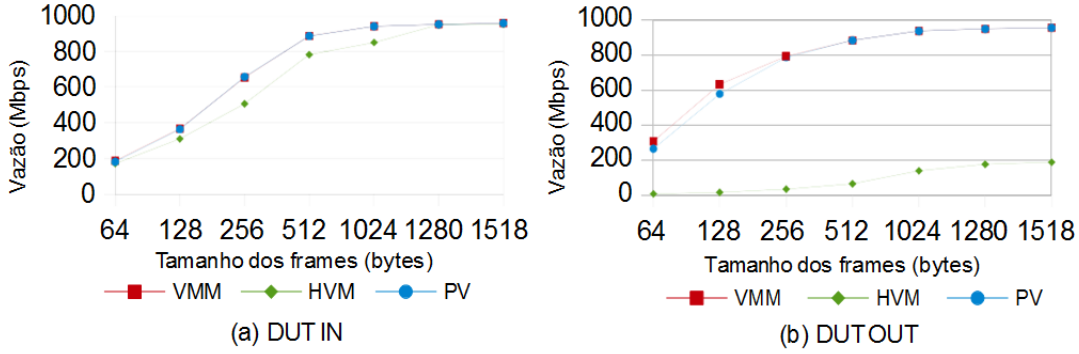


Figura 2. Taxa de vazão ou throughput

Quanto ao uso da CPU na recepção de dados (DUT IN), ambas tiveram um resultado satisfatório, onde o uso de CPU se manteve abaixo de 70% de processamento durante os testes de vazão. A Figura3(a) mostra os resultados.

Já no envio de dados (DUT OUT), observado na Figura 3(b), a VMM e a PV tiveram resultados semelhantes. A PV teve um pico máximo de 97.04% de uso nos *frames* de 64 *bytes*. A HVM não teve resultado satisfatório, pois o Qemu não foi projetado para ser *multithreading* na tradução binária, acarretando alto consumo de processamento (99%) de uma única CPU e, por consequência, baixa vazão de dados.

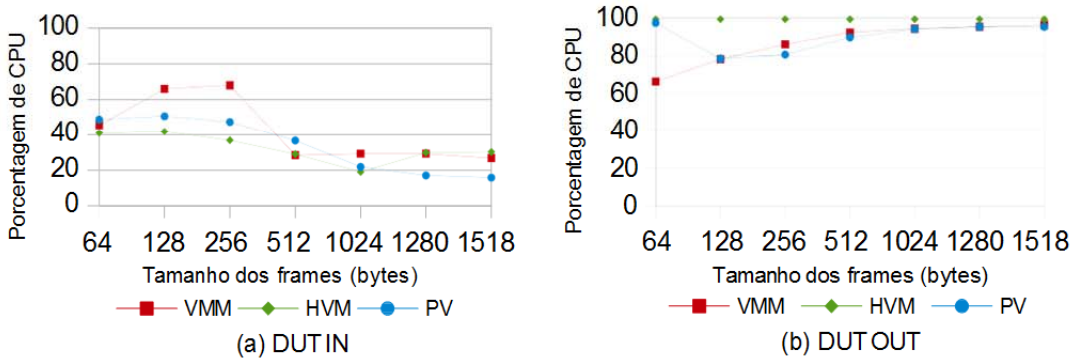


Figura 3. Desempenho da CPU

Na Figura 4(a) observa-se a taxa de perda de *frames* durante o recebimento de dados (DUT IN) nos experimentos de vazão. O HVM apresentou alta perda nos *frames* de 64, 128, 256, 512 e 1024 bytes, com pico de 23,33 % de erros em *frames* de 256 bytes, porém apresentou bons resultados nos frames de 1280 e 1518 bytes, com 0.1966% e 0,55% de perdas respectivamente. A paravirtualizada conseguiu atingir bons níveis de vazão com baixa perda de *frames* (pico de 2,5% de perda com *frames* de 64 *bytes*). A perda de *frames* na máquina física pode ser considerada desprezível por conta da alta vazão alcançada.

A Figura 4(b) representa a perda de *frames* durante experimentos (DUT OUT). Demonstra que a PV apresenta um pico na taxa de erros em *frames* de 128 bytes, com 3,37% de perda. A HVM não apresentou perda nos experimentos OUT devido a baixa vazão de dados ocasionado pela limitação de uso de uma única virtual CPU nas operações de network I/O. A taxa de perda da VMM teve pico de 0.378% com *frames* de 128 *bytes,* sendo considerado muito baixo.

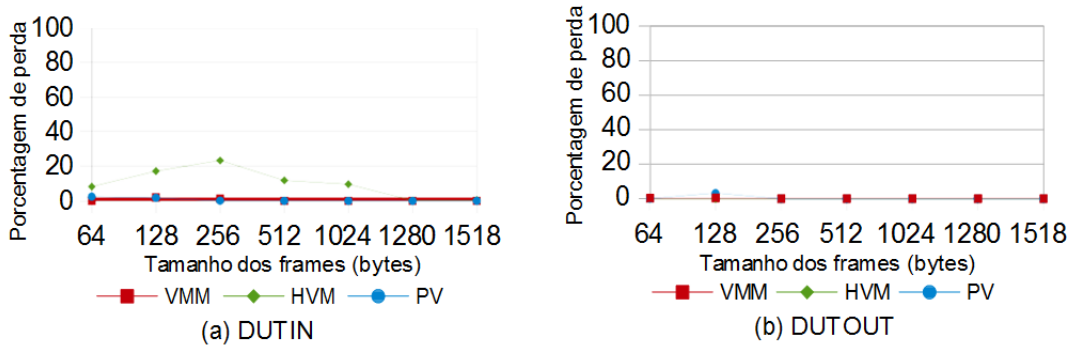


Figura 4. Perda de *frames*

Os experimentos de latência indicam que a camada de virtualização traz um acréscimo no tempo de resposta de processamento dos *frames* pelas VMs. Isso fica evidente na HVM, em que a latência é quase o dobro do VMM em alguns casos, novamente por conta do *driver* de rede emulado. A Figura 5 apresenta os resultados.

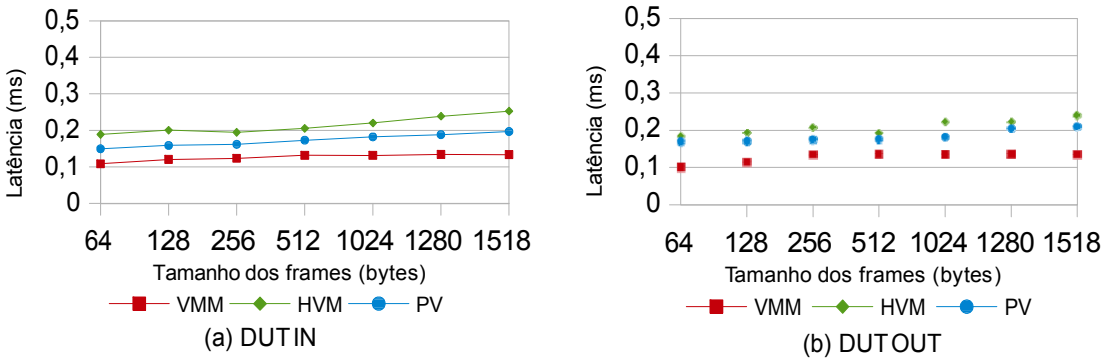


Figura 5. Latência

O principal problema da virtualização total é seu desempenho fraco na conversão binária. Acelerar a conversão binária é difícil. Segundo [Kiyanovski 2016] há quatro razões para o *driver* emulado e1000 (usado neste experimento) ser mais lento que o *driver* Virtio: 1. Ele quebra o segmento TCP/UDP para o tamanho dos *frames* MTU; 2. Ele calcula o TCP/UDP *checksum* para cada segmento; 3. Copia os dados do *guest* para o *host* antes de enviá-los; 4. Ele lida com interrupções de coalescência (combinação de partições adjacentes de memória livre) deficientemente para cenários de *throughput*.

Na paravirtualização, o *driver* de rede Virtio deixa a quebra do segmento e cálculo do *checksum* para o *host* físico, o qual executa ambas as tarefas muito mais rapidamente. O Virtio *driver* não executa a cópia dos dados, uma vez que os *buffers* não serão alterados até eles serem manipulados pelo dispositivo. E finalmente, cenários de *throughput* provocam interrupções a uma taxa consideravelmente inferior a emulação e1000, reduzindo a sobrecarga causada por interrupções.

6. Conclusão e Trabalhos Futuros

Visando reduzir custos e a complexidade dos ambientes de TI, a virtualização é uma tecnologia essencial para consolidação de serviços de redes, *datacenters*, NFV e computação em nuvem. Este experimento mostra a comparação de desempenho de rede em *link* Gigabit do VMM e máquinas virtuais com KVM, ambas com recursos da tecnologia Intel VT. As métricas avaliadas foram: *throughput*, perda de *frames*, latência e utilização de CPU. A RFC 2544 foi usada para validar as características de desempenho.

Os resultados mostram que a paravirtualização apresenta melhor desempenho e possui resultados muito similares ao *host* físico nas taxas de vazão recebidas e enviadas, perda de *frames* e latência. A virtualização assistida por *hardware* apresentou um bom desempenho nas taxas de vazão recebidas e não satisfatório nas taxas envidadas.

A Tecnologia Intel VT foi desenvolvida para otimizar o desempenho das máquinas virtuais, mas os ganhos se aplicam somente ao uso dos recursos do

Processador. O *throughput* de rede em dispositivos emulados não apresenta ganho de desempenho significativo, devido a limitação de uso de vCPU pelo Qemu e seu desempenho fraco na tradução binária. As principais razões pela lentidão na tradução binária são a quebra do segmento TCP/UDP em *frames* e o cálculo de *checksum* ao envidar dados. A paravirtualização apresenta melhor desempenho, porém a máquina virtualizada precisa ser modificada para ser compatível, podendo ocasionar problemas de compatibilidade e portabilidade.

Para qualquer sistema operacional moderno, os *drivers* gerenciados pelo Qemu só deveriam ser usados provisoriamente como um suporte para colocar o sistema em funcionamento até que se possa instalar e mudar para os Virtio *drivers,* em casos que não possam ser instalados juntamente com o Sistema Operacional. Observa-se que otimizar *drivers* emulados para executar em múltiplas vCPU não é prioridade dos desenvolvedores do Qemu, por isso recomenda-se mudar para os Virtio *drivers* caso não haja problemas de compatibilidade e portabilidade.

Trabalhos futuros incluem estudos para diminuir a diferença entre a paravirtualização e a virtualização total, abordando o principal problema da virtualização encontrada neste experimento, a conversão binária fraca.

Referências

Carvalho, F. L. and Bellezi, M. A. (2014). Avaliação de Desempenho dos Hypervisors XEN e KVM utilizando Administração Simplificada através do LibVirt. T.I.S. São Carlos, v. 3, n. 1, p. 88-101.

Costa, G. H. Da. (2008). Métricas para Avaliação de Desempenho em Redes QoS sobre IP. Curso De Especialização Em Tecnologias, Gerência E Segurança De Redes De Computadores. UFRGS.

Grinberg, S. and Weiss, S. (2012). Architectural virtualization extensions: A systems perspective. Computer Science Review. November 2012, Vol.6(5-6), pp.209-224.

Kiyanovski, A. (2016). Student research poster: Network controller emulation on a sidecore for unmodified virtual machines. Parallel Architecture and Compilation Techniques (PACT), 2016 International Conference on.

Kolling, M. L. et al. (2008). Verificação em Hardware de Componentes de Comunicação. IX Simpósio em Sistemas Computacionais. Departamento de Informática. Universidade de Santa Cruz do Sul (UNISC). p. 143-150.

Motika G. and Weiss S. (2012). Virtio Network paravirtualization driver: Implementation and performance of a de-facto standard. Computer standards &

interfaces, vol. 34, pg. 36-47. Elsevier.

Patel, M. and Chaudhary, S. (2014) Survey on a combined approach using prediction and compression to improve pre-copy for efficient live memory migration on Xen. Parallel, Distributed and Grid Computing (PDGC), 2014 International Conference on, pages 445-450. IEEE.

Shirinbab, S., Lundberg, L. and Ilie, D. (2014) Performance Comparison of KVM, VMware and XenServer using a Large Telecommunication Application. The Fifth International Conference on Cloud Computing GRIDS and Virtualization.

Módulo Didático com Tecnologia VLC

Wagner V. Longhi[1], Claiton P. Colvero[1]

[1]Colégio Técnico Industrial de Santa Maria – Universidade Federal de Santa Maria (CTISM - UFSM) – Av. Roraima, 1000 – 97.105-900 – Santa Maria – RS - Brasil

{wagner.longhi,claiton}@redes.ufsm.br

Abstract. *Due to the current saturation of the electromagnetic spectrum in the radiofrequency bands for wireless communication, it is necessary to use new frequencies for these services. This paper describes the study and implementation of a free space optical communications (FSOC) didactic module, which uses wavelengths of the visible spectrum to offer non-atmosphere-guided links known as VLC (Visible Light Communication). The developed module can be used for the learning environment to complement the didactic contents in the classroom, it is allowing easy visualization of the effects that are occurring in the propagation of the signals in an intuitive and simplified method.*

Resumo. *Devido a atual saturação do espectro eletromagnético nas faixas de radiofrequência para as comunicações sem fio, surge a necessidade de utilização de novas frequências para os mesmos serviços. Este artigo descreve o estudo e implementação de um módulo didático de comunicações ópticas no espaço livre (FSOC), que utiliza os comprimentos de onda do espectro visível para implementar enlaces não guiados pela atmosfera, conhecido como VLC (Visible Light Communication). O módulo desenvolvido pode ser utilizado no ambiente de ensino e aprendizado para complementar os conteúdos ministrados em sala de aula, permitindo a fácil visualização dos efeitos que estão ocorrendo durante a propagação dos sinais de forma intuitiva e simplificada.*

1. Introdução

O aumento na demanda e a grande competitividade do mercado na área de telecomunicações incentiva a busca por novas tecnologias e sistemas que possam suprir carências do setor, como o caso específico da falta de disponibilidade de frequências livres no espectro eletromagnético para as comunicações sem fio convencionais [Anatel, 2012]. Atualmente tem-se observado uma disputa intensa por faixas de frequências disponíveis para a implementação de novos produtos e serviços de telecomunicações. A partir destas observações, pesquisas e estudos tem se desenvolvido para oferecer modernas técnicas de comunicação de banda larga em uma faixa do espectro eletromagnético menos saturada, onde destacam-se as comunicações ópticas no espaço livre FSOC e VLC [Filho, 2015].

Para o desenvolvimento deste trabalho, inicialmente foi realizada uma pesquisa sobre as novas tecnologias de comunicações ópticas não guiadas por fibra, que permitiu projetar e implementar de um módulo didático de baixo custo para a utilização nas disciplinas de telecomunicações do curso de Redes de Computadores do CTISM (Colégio Técnico Industrial de Santa Maria). As pesquisas foram realizadas com base nas propostas de novas tecnologias e produções acadêmicas mais recentes, tendo em vista que estes sistemas ainda possuem baixa disponibilidade comercial, o que limita as fontes mais aplicadas de estudo.

Complementando o módulo VLC didático deste trabalho, também foi desenvolvido um aplicativo intuitivo com diferentes funcionalidades e experimentos direcionados para o aprendizado da tecnologia em laboratório. Para a verificação das funcionalidades e limitações do sistema foram realizados diferentes ensaios com condições controladas no laboratório de telecomunicações do CTISM, confrontando os resultados das medições com os efeitos observados visualmente, uma vez que foram utilizados comprimentos de onda da luz visível para o melhor entendimento dos alunos.

2. Protótipo

Nesta seção são demonstradas as tecnologias selecionadas para operação do protótipo inicial do módulo, os materiais e os métodos utilizados para o desenvolvimento do mesmo. O detalhamento das atividades e montagem dos dispositivos utilizados está descrito na ordem cronológica de desenvolvimento, de forma sucinta e objetiva para o melhor entendimento sobre os processos de execução.

2.1. Módulo Arduino UNO R3

A utilização desta plataforma de prototipagem de hardware foi motivada pelo baixo custo de aquisição da mesma sem comprometer a operação do sistema como um todo. Através da utilização do aplicativo Arduino IDE, foram realizados o desenvolvimento e a gravação dos códigos na linguagem de programação C em seu micro controlador [McRoberts, 2010]. Outra vantagem desta plataforma é a possibilidade de alimentação através da mesma entrada onde é realizada a comunicação com o computador, eliminando a necessidade de utilização de uma fonte de energia externa.

Nas duas interfaces Arduino utilizadas, uma para transmitir os sinais e a outra para receber, foi necessário desabilitar um recurso chamado *auto reset*, para que não ocorra uma reinicialização das placas sempre que realizada uma comunicação serial. Para desabilitar esse recurso, foi inserido um capacitor eletrolítico de 10 μF entre os pinos GND e RES.

2.1.1. Módulo Transmissor

O Arduino utilizado como transmissor executa o tratamento dos caracteres enviados pela interface transmissora do aplicativo, via comunicação serial. Essa informação recebida é tratada e convertida para uma sequência binária para depois ser enviada através da modulação OOK-NRZ (*non-return-to-zero on-off-keying*) [Kartalopoulos, 2003]. A energia luminosa é gerada por um dispositivo LED (*Light Emitting Diode*) conectado na saída PWM (*Pulse Width Modulation*) do Arduino, de acordo com o bit enviado pelo sistema.

Quando energizado, o código gravado no micro controlador aguarda a recepção do tempo de transmissão por bit, o qual é digitado pelo usuário na interface transmissora. Para alterar o tempo de transmissão por bit, é necessário realizar uma reinicialização manual do Arduino e reexecutar o aplicativo, que define a interface de transmissão.

Um protocolo simplificado foi desenvolvido e inserido no código da interface para realizar a tradução, modulação e inserção de balizas de sinalização nas mensagens que são enviadas. Nesse protocolo, cada caractere é representado por uma sequência binária de seis dígitos. A sequência “101010” é utilizada para sinalizar o início da transmissão, onde o primeiro bit é o *start* bit e os outros cinco bits são utilizados para o receptor verificar se a mensagem a ser recebida é, de fato, a mesma enviada pelo transmissor. A sequência “000000” sinaliza o fim da transmissão, sobrando livres 62 posições para envio de diferentes caracteres.

Complementando a interface de envio das informações existe a torre transmissora (Figura 1), formada por um LED, uma lente no modelo plano convexa, um cilindro onde as peças são acomodadas, a haste de ajuste do ponto focal e uma base articulável. O LED utilizado para gerar o feixe de luz do sistema de transmissão, emite um feixe luminoso com o comprimento de onda da luz visível variando entre 615 nm e 625 nm [Valadares et al., 2009]. A lente plano convexa tem a função de diminuir a dispersão dos fótons, colimando o feixe luminoso [Ortiz, 1862, p.307-311] emitido pelo LED.

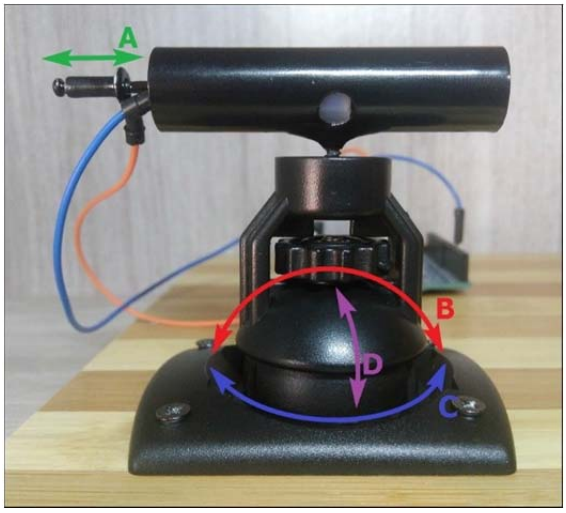


Figura 1. Torre Transmissora com o ajuste de ponto focal em 'A' e ajustes de direcionamento do feixe luminoso em 'B', 'C' e 'D'.

A torre transmissora foi montada utilizando componentes de diferentes objetos para minimizar os custos do projeto. O cilindro foi obtido de uma caneta laser, para a haste de ajuste foi utilizado um rebite, e a base foi obtida de um suporte de GPS veicular. Mesmo com um baixo custo para a produção deste componente, este permite um direcionamento preciso para o apontamento com a torre receptora que está a uma distância de 24 centímetros.

2.1.2. Módulo Receptor

O Arduino utilizado no receptor decodifica os sinais ópticos detectados por um fotodiodo para sequências binárias recebidas, converte essas sequências para caracteres ASCII e faz o envio da mensagem para a interface receptora, através de comunicação serial.

Assim como no Arduino transmissor, inicialmente o código aguarda a recepção do tempo de transmissão por bit, digitado pelo usuário na interface receptora. Para manter o sistema sincronizado, é importante informar o mesmo tempo nos dois módulos Arduino.

No receptor, o protocolo tem a função de verificar, através de uma comparação, quando a sequência binária sinalizadora de início é recebida. Então é iniciado o processo de tradução para caracteres até que seja detectada a sequência binária que sinaliza o fim da transmissão. Durante a tradução, a cada seis bits recebidos, é feita a conversão para o caractere correspondente e esse é armazenado em uma variável do tipo *string*.

O código foi desenvolvido para realizar a leitura do nível de tensão na entrada analógica do Arduino, para obter os níveis de sinal alto, baixo ou de limite para troca de bit. Esse nível é convertido para um valor lógico de 10 bits de amostragem que pode variar de 0 até 1023 unidades arbitrárias. Neste protocolo também é definido um limite

para troca de estado do bit, onde o receptor entende como bit ‘0’ um valor lógico maior do que zero, mas abaixo do limite estabelecido, e como bit ‘1’ o valor lógico acima do limite de decisão do decodificador. A entrada digital do Arduino também poderia ser utilizada neste projeto, mas a falta de transmissão (inatividade) poderia eventualmente ser confundida com um nível zero, o que não resultaria em uma decodificação correta.

Na interface de recepção do sinal fazem parte o módulo Arduino, a torre receptora (Figura 2) e o amplificador operacional. Esta torre é composta por um fotodiodo, um cilindro de foco, uma haste de ajuste e uma base articulável. O fotodiodo faz a conversão dos fótons que incidirem na sua superfície para um valor correspondente de corrente elétrica [Su et al., 2011]. Um amplificador operacional configurado com realimentação negativa [Pertence, 2015], eleva os níveis de tensão para um valor compatível com a entrada analógica do Arduino. O cilindro e a haste limitam o FOV (*Field of View*) do fotodiodo, para que ruídos causados por fontes de luz externas não interfiram na recepção.

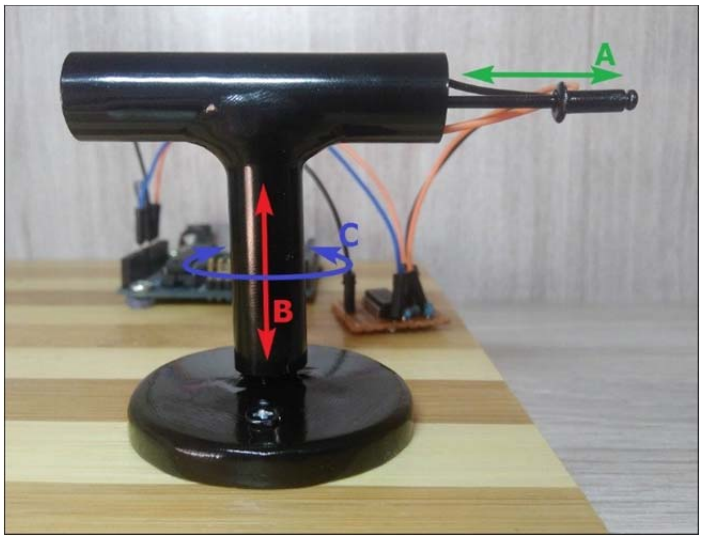


Figura 2. Torre Receptora com ajuste do FOV em ‘A’ e ajustes de direcionamento em ‘B’ e ‘C’.

A base da torre receptora também foi fabricada com resina epóxi, recebendo componentes reaproveitados de diferentes objetos para manter o baixo custo na produção. O ajuste de altura da torre indicado por ‘B’ (Figura 2) foi realizado a partir de uma seringa, que sofreu algumas modificações para se adequar a estrutura da torre.

2.2. Interfaces de Comunicação

As interfaces de comunicação foram projetadas para proporcionar uma melhor interatividade do usuário com os módulos receptor e transmissor deste trabalho. Para o desenvolvimento destas interfaces foram utilizadas a linguagem de programação *Shell Script* [Jargas, 2008], com o *bash* como interpretador de comandos, e a ferramenta *Zenity*, para gerar as interfaces de diálogo onde é realizada a interação com o usuário. Inicialmente o código desenvolvido é responsável por detectar automaticamente a porta onde está conectada a interface Arduino, então é exibida uma caixa para inserção de texto, onde o usuário pode digitar o tempo de transmissão por bit. Após esta entrada é exibido ao usuário um menu de opções, onde é possível acessar o item de ajuda, que disponibiliza uma caixa de texto com explicações detalhadas de como operar o sistema e executar suas funcionalidades.

2.2.1. Interface Transmissora

A interface transmissora possui a função de detectar a porta do computador onde o Arduino transmissor está conectado, registrar o tempo de transmissão por bit, e registrar a mensagem digitada pelo usuário. Entre as opções exibidas no menu da interface, é encontrada a opção "Digitar_mensagem", que carrega uma caixa de texto para o usuário digitar a mensagem a ser enviada pela tecnologia VLC até o receptor.

2.2.2. Interface Receptora

A interface receptora inicialmente detecta a porta em que o Arduino receptor está conectado e registra o tempo de transmissão por bit digitado pelo usuário. Após é exibido um menu de opções, onde pode ser acessada, entre outras, a opção "Exibir_mensagens", para carregar e exibir as mensagens já recebidas, ou a opção "Limpar_log", para apagar as mensagens já recebidas.

3. Resultados

Nesta seção são apresentados os resultados mais relevantes obtidos nos ensaios de laboratório em condições controladas. Foram realizados diferentes ensaios com o sistema VLC desenvolvido para a coleta dos primeiros resultados de operação em diferentes condições de propagação dos sinais ópticos pela atmosfera, definindo as suas capacidades e limitações específicas. Também são sendo listados os problemas encontrados nos ensaios de operação e as respectivas soluções implantadas para contornar essas falhas, mantendo o protótipo com a maior eficiência possível.

3.1. Ensaio em Ambiente sem Iluminação – Ruído de Fundo

Nesse ensaio inicial, o protótipo foi colocado em operação em um ambiente onde não existia nenhum tipo de iluminação externa para evitar a contaminação de fótons que não pertenciam ao sistema. O objetivo deste ensaio foi avaliar a dinâmica de recepção do fotodetector utilizado em um ambiente controlado para a calibração dos valores resultantes na operação real, conforme apresentado para o ambiente escuro (Tabela 1).

Tabela 1. Resultados obtidos com o ensaio em um ambiente sem iluminação externa – Avaliação do ruído de fundo do sistema para calibração

Tempo por bit (ms)	4	8	16	32	64
Número máximo de caracteres	4	10	21	41	75
Taxa de bits (bps)	250	125	62,5	31,2	15,6
Nível de sinal baixo (0 – 1023 u.a.)	0				
Limite de nível (0 – 1023 u.a.)	100				
Nível de sinal alto (0 – 1023 u.a.)	250				

Os resultados da tabela demonstram na 1º linha o tempo padrão utilizado para o envio de um bit, na 2º linha o número máximo de caracteres sem que ocorra algum erro na transmissão, na 3º linha a taxa de transmissão máxima que o sistema atinge, na 4º linha o valor de tensão que representa bit 0 lido na entrada analógica do Arduino, na 5º linha o valor lógico definido para o sistema diferenciar as leituras de bit 1 e bit 0, e na 6º linha o valor de tensão lido para o bit 1. Os valores de tensão lidos foram convertidos em níveis lógicos de 10 bits (0 – 1023 u.a.). Para as demais tabelas deste trabalho, a 3º linha (taxa de transmissão) está sendo omitida para diminuir a redundância das informações.

3.2. Transmissão de Sinais Através da Água - Atenuação

Para aproximar a operação do sistema simulando a transmissão de sinais ópticos em um ambiente chuvoso, e considerando que os feixes de luz atingirão preferencialmente o eixo central das gotículas de chuva, foi inserido entre as torres transmissora e receptora um recipiente com um formato que se aproxima de um quadrado, cheio de água, por onde os fótons devem atravessar para chegar até o fotodiodo.

Os resultados (Tabela 2) foram semelhantes aos apresentados no ensaio anterior (Tabela 1), com uma diferença visível nos níveis de sinal alto e baixo. Devido aos ruídos causados pela presença de luz natural no ambiente, estes níveis se elevaram e necessitaram de uma calibração posterior. A comunicação ocorreu sem a presença de efeitos de refração no sinal, pois seu eixo de propagação é perpendicular à parede do recipiente, não alterando a trajetória do feixe pela mudança do índice de refração dos meios (ar – água – ar), semelhante ao que aconteceria ao atravessar uma gota de chuva pelo eixo central.

Tabela 2. Resultados obtidos com a transmissão de sinais através da água

Tempo por bit (ms)	4	8	16	32	64
Número máximo de caracteres	4	10	21	41	78
Nível de sinal baixo (0 – 1023 u.a.)	5				
Limite de nível (0 – 1023 u.a.)	100				
Nível de sinal alto (0 – 1023 u.a.)	260				

3.3. Recipiente com Água e Partículas - Espalhamentos

Neste experimento foi realizada uma diluição de 5% de leite no volume total de água do recipiente utilizado no ensaio de transmissão pela água. O objetivo deste experimento é avaliar a transmissão através de um meio com alta densidade de pequenas partículas em suspensão, diretamente no caminho do feixe de luz, que acarreta no redirecionamento e absorção dos fótons durante a transmissão. Pode-se observar pela superfície do recipiente que o feixe de luz ilumina as partículas do leite que está diluído na agua (Figura 3), alterando sua direção de propagação e não permitindo que este chegue até o receptor. Nesse caso, mesmo ajustando o valor do limite de nível, o Arduino receptor falhou diversas vezes na detecção, ocasionando na inversão da leitura ao ler bit “1” com bit “0” e vice-versa.

Figura 3. Visualização do feixe de luz devido à presença de partículas no meio.

3.4. Ensaio com Redirecionamento do Feixe Através de Espelhos

Com o auxílio de espelhos planos posicionados adequadamente para refletir o feixe luminoso algumas vezes, foi realizada a simulação da propagação dos sinais ópticos confinados, como

em uma fibra óptica ou em um mecanismo de multipercurso. Embora este último caso não possa ser considerado uma forma guiada de transmissão dos sinais ópticos, ele representa uma característica comum de propagação do feixe em um ambiente real de utilização, sendo este direcionado de acordo com os elementos interferentes do entorno do enlace montado.

Conforme demonstrado nos resultados obtidos (Tabela 3), foi necessário alterar o limite de nível para o valor 30, devido à queda dos limiares dos sinais recebidos pela atenuação de fótons durante a reflexão nos espelhos, e devido também a distância do percurso que passou de 24 para 35 centímetros. A 6º coluna da 2º linha representa que foram transmitidos somente 36 caracteres sem a ocorrência de erros utilizando 64 ms por bit, a ocorrência prematura de erros se deu devido à proximidade entre o nível de sinal baixo com o nível de sinal alto. Também foi necessário regular a haste da torre receptora, resultando no aumento do ângulo de visão do fotodiodo.

Tabela 3. Resultados obtidos com o redirecionamento do feixe luminoso

Tempo por bit (ms)	4	8	16	32	64
Número máximo de caracteres	4	10	21	42	36
Nível de sinal baixo (0 – 1023 u.a.)	0				
Limite de nível (0 – 1023 u.a.)	30				
Nível de sinal alto (0 – 1023 u.a.)	44				

3.5. Transmissão de Sinais Através da Fumaça

Nesse ensaio o ambiente de transmissão foi preparado para simular um fenômeno de nevoeiro (condensação da água e partículas sólidas em suspensão). Como se trata de um experimento didático em laboratório, na preparação, foi realizada a queima de carvão vegetal para gerar o espalhador da luz, representando o nevoeiro. Uma grande parte do feixe luminoso foi refletido ao tentar atravessar a fumaça, o que causou forte atenuação no sinal.

Os resultados obtidos (Tabela 4) demonstram que devido à forte atenuação do sinal, o limite de nível teve de ser ajustado para manter a relação entre o nível de sinal baixo e alto. A leitura obtida do nível de sinal baixo, encontrada na 3º linha, foi 0, pois não havia nenhum ruído no ambiente durante a realização do experimento. Na 5º linha é exibido o valor lógico 94, com uma regra de três é possível identificar que ocorreu uma atenuação de 0,76 volts no sinal recebido em comparação com o primeiro experimento realizado (Tabela 1).

Tabela 4. Resultados obtidos com a transmissão através da fumaça

Tempo por bit (ms)	4	8	16	32	64
Número máximo de caracteres	4	10	21	41	78
Nível de sinal baixo (0 – 1023 u.a.)	0				
Limite de nível (0 – 1023 u.a.)	50				
Nível de sinal alto (0 – 1023 u.a.)	94				

4. Conclusão

Este trabalho desenvolveu um módulo didático com tecnologia VLC para utilização em processos de ensino e aprendizado nos laboratórios de telecomunicações, com objetivo de demonstrar de forma prática e intuitiva os fenômenos envolvidos na propagação de sinais ópticos pela atmosfera. Através da utilização de comprimentos de onda de luz visível e experimentos direcionados para a simulação dos efeitos naturais de propagação, estes

módulos apresentaram significativa aplicabilidade e eficiência no complemento dos ensinamentos de sala de aula.

Foram realizados diferentes experimentos em condições controladas para a simulação de operação em ambientes reais de utilização, confrontando os resultados obtidos através das medições com a visualização dos traçados do feixe de luz visível, conferindo ao sistema uma boa coerência em relação aos resultados esperados. Pequenos problemas foram encontrados e resolvidos a partir dos experimentos realizados, incrementando a eficiência do sistema como um todo.

O protótipo já se encontra operacional, e está sendo realizada a implementação de modelos para impressão em 3D das estruturas e proteções. Este próximo passo do trabalho permite que sejam desenvolvidas diversas unidades educacionais para o CTISM em um curto período de tempo e baixo custo, permitindo uma maior utilização de todos os alunos do Curso de Redes de Computadores nas disciplinas desta área de conhecimento.

Referências

9TH EUROPEAN CONFERENCE ON RADIATION AND ITS EFFECTS ON COMPONENTS AND SYSTEMS, 2007, Deauville. BPW34 commercial p-i-n diodes for high-level 1-MeV neutron equivalent fluence monitoring... IEEE, 2009. 8 p. Disponível em: <http://ieeexplore.ieee.org/document/5205483>. Acesso em: 14 jun. 2017.

ANATEL, Plano de atribuição, destinação e distribuição de faixas de frequências no Brasil. Disponível em: http://www.anatel.gov.br/Portal/verificaDocumentos/documento.asp. Acesso em: 10 jun. 2017.

FILHO, Vilmey Francisco Romano. VLC - Comunicação Óptica por Luz Visível. Trabalho de Conclusão de Curso – Universidade de Brasília – UnB, Faculdade Gama - FGA, 2015.

GOMEZ, A.; KAI Shi; QUINTANA, C.; SATO, M.; FAULKNER, G.; THOMSEN, B.C.; O'BRIEN, D., Beyond 100-Gb/s Indoor Wide Field-of-View Optical Wireless Communications, Photonics Technology Letters, IEEE, vol.27, no.4, pp.367,370, 2015.

JARGAS, Aurélio Marinho. Shell Script profissional. São Paulo: Novatec, 2008. 481 p.

KARTALOPOULOS, S. V. DWDM: Networks, Devices, and Technology. 1. ed. Wiley-IEEE Press, 2002. 520 p. Disponível em: <http://www.globalspec.com/reference/21547>. Acesso em: 14 jun. 2017.

MCROBERTS, M. Arduino Básico. 2. ed. São Paulo: Novatec, 2015. 25 p. Disponível em: <https://s3.novatec.com.br/capitulos/9788575224045.pdf>. Acesso em: 22 jun. 2017.

ORTIZ, P. P. Óptica: Prismas y lentes. In: ORTIZ, P. P. Principios Elementales de Fisica Experimental Y Aplicada... Conteniendo Todos Los Ultimos Descubrimientos Y Aplicaciones Recientes a la Industria, Artes, Etc., Usos Y Objetos de la Vida Comun. New York: Columbia University, 1862. cap. 14, p. 307-311. Disponível em: <https://books.google.com.br/books?id=MVVDAAAAYAAJ>. Acesso em: 14 jun. 2017.

TANENBAUM, A. S.: Redes de Computadores. 4ª Ed., Editora Campus (Elsevier), 2003.

VALADARES, Eduardo C.; MOREIRA, Alysson M. Caderno Catarinense de Ensino de Física. Ensinando física moderna no segundo grau: efeito fotoelétrico, laser e emissão de corpo negro, Belo Horizonte, p. 121-135, ago. 2009. Disponível em: <https://periodicos.ufsc.br/index.php/fisica/article/view/5205483>. Acesso em: 08 jun. 2017.

Análise de Estratégias que Fazem Uso de Informações da Estrutura Topológica para o Posicionamento de Nós Regeneradores em Redes Ópticas Translúcidas

Nilvan Santana Souza, Ueslem de Oliveira Pereira, Gilvan Martins Durães

Instituto Federal Baiano (IF Baiano) – Campus Catu R. Barão de Camaçari, 118, Centro - 48110-000 – Catu – BA – Brasil

nilvan_asouza@hotmail.com, ueslempereira@gmail.com, gilvan.duraes@catu.ifbaiano.edu.br

Abstract. *Optical networks have been proposed as the technology that can support the increasing demand of Internet traffic in contemporary society. In the optical network, the optical signal suffers physical impairments due to the distance that it travels, for example. So, in some cases, this light must be regenerated by an Optical-Electrical-Optical (OEO) converter. In this context, there is a need to answer the following questions: where it to placement an OEO regenerator? How many OEO regenerators are necessary to ensure the quality of network connections? In this way, heuristics have been proposed to the regenerator's allocation. This work proposes two new strategies to regenerator's allocation that use information from network's topology. One of proposed strategies showed better performance in terms of the number of regenerators nodes required and the blocking probability of optical connections when compared to other equivalent strategies from the literature, in all the evaluated scenarios.*

Resumo. *As redes ópticas têm sido propostas como a tecnologia que pode atender a alta demanda de tráfego da Internet na sociedade contemporânea. Na rede óptica o sinal de luz sofre degradações na fibra, entre outros fatores, devido à distância que ele percorre, então em alguns casos esse sinal de luz precisa ser regenerado por um conversor chamado Óptico-Elétro-Óptico (OEO). Nesse contexto, existe a necessidade de responder às seguintes perguntas: onde posicionar um regenerador OEO? Qual a quantidade necessária para garantir a qualidade das conexões na rede? Desta forma, heurísticas têm sido propostas para a alocação desses regeneradores. Este trabalho propõe duas novas estratégias para o posicionamento de nós regeneradores que utilizam informações da estrutura topológica da rede. Uma das estratégias propostas apresentou melhor desempenho em termos de quantidade de nós regeneradores requeridos e outra em termos de probabilidade de bloqueio de conexões ópticas quando comparado com outras estratégias equivalentes da literatura, em todos os cenários avaliados.*

1. Introdução

A Internet se tornou indispensável para vida humana, o número de usuários da Internet tem crescido a cada dia e consequentemente há o aumento da demanda por banda nas redes de transporte localizadas no núcleo da Internet. Segundo o décimo Relatório *Cisco® Visual Networking Index (VNI) Global Forecast and Service Adoption* 2014-2019 [Cisco 2015], o tráfego IP global anual superou os Zettabytes (1000 Exabytes) em 2016, e irá superar dois zettabyte em 2019, ou seja, a previsão é que o tráfego da

Internet dobre de volume nesse período. Para atender a essa demanda crescente de tráfego na Internet se justifica o uso das Redes Ópticas, pois a mesma apresenta grande largura de banda, podendo transportar os dados em alta velocidade [Soares *et al.* 2012].

Segundo [Fontinele *et al.* 2014], a medida em que o sinal se propaga na rede, ele vai perdendo sua qualidade, e os próprios equipamentos da rede inserem degradações no sinal óptico. Quando não há regeneração do sinal, as degradações vão acumulando e consequentemente diminuindo a qualidade do sinal, podendo chegar a níveis intoleráveis. Assim, efeitos de camada física, como ruído do amplificador, dispersão, *crosstalk* e efeitos não-lineares, são exemplos de degradações causadas pelos componentes da rede [Zhao *et al* 2014]. Desta forma, a qualidade do sinal deve ser mantida em níveis satisfatórios, se isso não ocorrer, a conexão poderá ser bloqueada. Assim, as redes ópticas podem ser classificadas pela capacidade que seus nós têm em regenerar o sinal óptico, são elas: redes ópticas opacas, redes ópticas transparentes e redes ópticas translúcidas.

Na rede opaca todos os nós possuem um conversor chamado Óptico-Elétro-Óptico (OEO), o mesmo converte o sinal óptico em eletrônico e, em seguida, depois de processado, ele é novamente convertido em óptico, para que cada nó da rede garanta a qualidade do sinal. Contudo esses equipamentos inserem atrasos no processamento da rede e aumento no custo dos equipamentos [Fontinele *et al.* 2016].

Na rede transparente os nós não possuem a capacidade de regenerar o sinal óptico, isto é, o sinal trafega pelos nós da rede somente em domínio óptico, não há regeneração do sinal. Consequentemente, a rede terá mais velocidade no estabelecimento de conexões, mas sem a regeneração do sinal surge o problema da degradação de camada física [Nath *et al* 2014].

Já na rede translúcida, apenas alguns nós possuem o conversor Óptico-Elétro-Óptico, ou seja, alguns nós da rede são selecionados estrategicamente para possuírem conversores que irão regenerar o sinal óptico. Esse tipo de rede busca agregar a velocidade da rede transparente com a qualidade do sinal da rede opaca [Nath *et al* 2014], [Durães *et al.* 2016].

Desta forma, este trabalho consiste em investigar onde posicionar o regenerador OEO na rede e quantos deles serão necessários para que o sinal esteja em índices de qualidade aceitáveis nas conexões entre todos os pares de nós (origem, destino) da rede, visando também ao menor atraso possível do sinal óptico em toda a rede. Para isso, este trabalho analisa o desempenho de estratégias de dimensionamento e posicionamento de nó regenerador OEO baseadas apenas em métricas que utilizam apenas informações topológicas da rede [Nath *et al* 2014].

Sendo assim, foram analisadas as estratégias *Nodal Degree First* (NDF) e *Hub Node First* (HNF) [Nath *et al* 2014], além de estratégias baseadas na Centralidade de Intermediação (*Betweenness*) [Freitas 2010] e Distância Média [Guimaraes *et al* 2015]. Essas estratégias consistem em heurísticas de dimensionamento e posicionamento de nós regeneradores que utilizam informações topológicas da rede como parâmetro. Neste trabalho, as estratégias citadas foram analisadas sob as seguintes métricas: Número de regeneradores e Probabilidade de Bloqueio de requisições de conexões ópticas.

O restante do artigo está organizado da seguinte forma: na Seção 2, é apresentado o problema de Posicionamento de Nós Regeneradores. Na Seção 3, são apresentados os trabalhos relacionados e as principais contribuições deste artigo. Na

Seção 4, os resultados de avaliação de desempenho são apresentados e discutidos. Por fim, na Seção 5, são apresentadas as considerações finais e trabalhos futuros.

2. Problema de Posicionamento de Nós Regeneradores

Em redes ópticas translúcidas usa-se um conjunto de regeneradores posicionados estrategicamente com o propósito de regenerar a qualidade do sinal óptico e diminuir os custos da rede. Existem várias soluções heurísticas que são utilizadas para resolver esse problema baseadas em informações da topologia, do tráfego ou da utilização da rede [Soares *et al* 2012], [Nath *et al* 2014]. Assim, o objetivo do posicionamento de regeneradores é minimizar os impactos das degradações da camada física óptica.

As estratégias consideradas neste trabalho, as quais utilizam apenas informações da topologia da rede como parâmetro, constituem o grupo das estratégias mais simples para este tipo de problema [Nath *et al* 2014]. Nota-se que esse problema é tratado na fase de planejamento da rede, sendo que os custos e restrições da camada física óptica devem ser considerados para a definição da quantidade de regeneradores [Nath *et al* 2014].

Desta forma, as estratégias que utilizam informações da estrutura topológica funcionam da seguinte forma: posiciona-se um regenerador após o outro no nó transparente que possui maior importância em relação aos demais. Essa importância é definida pelo algoritmo de posicionamento, em função, por exemplo, da métrica maior grau (número de enlaces físicos ou lógicos) ou centralidade do nó. Portanto, esse procedimento é repetido para o demais nós da rede que não possuem regenerador até que seja possível todos os pares de nós origem e destino se alcancem, respeitando uma restrição física de topológica. Nos experimentos apresentados neste trabalho, a restrição física considerada na fase de planejamento da rede foi dois saltos transparentes, ou seja, a cada dois saltos, um caminho óptico deve passar por um nó regenerador OEO.

Este trabalho considera uma rede óptica translúcidas com posicionamento esparso de nós OEO, onde somente uma parte de nós na rede possui capacidade de regeneração, porém esses nós podem regenerar o sinal de qualquer conexão que passa por ele [Soares *et al* 2012], [Nath *et al* 2014]. O posicionamento de nós regeneradores influencia diretamente na qualidade do sinal óptico das conexões ativas da rede e, consequentemente, na Probabilidade de Bloqueio de requisições de conexões.

3. Trabalhos Relacionados e Contribuições

Considerando apenas as informações da topologia da rede, foram utilizadas as seguintes heurísticas de posicionamento de nós OEO propostas ou adaptadas da literatura: *Nodal Degree First* (NDF) [Yang e Ramamurthy, 2015] e *Hub Node First* (HNF) [Shen e Grover 2004], e estratégias baseadas nas métricas Centralidade dos nós (*Betweennness*) [Freitas 2010] e Distância Média [Araújo *et al.* 2014], [Guimaraes *et al* 2015]. Além dessas estratégias heurísticas, foram propostos dois novos algoritmos de posicionamento, chamados *Impairments Aware - Nodal Degree First* (IA – NDF) que considera maiores informações da camada física e *Modified Hub Node First* (M-HNF).

O algoritmo NDF, proposto em [Yang e Ramamurthy, 2015] seleciona como nó regenerador aquele que possui o maior grau nodal (número de interfaces físicas/número de links), se houver mais de um nó com o mesmo grau físico essa escolha é feita de forma aleatória entre eles. Na medida em que o algoritmo escolhe um nó da rede como nó regenerador, o grau nodal de todos os seus vizinhos são diminuídos.

Já, o algoritmo HNF [Shen e Grover 2004] prioriza os nós que possuem maior grau lógico para tornar-se um nó regenerador. O grau lógico de um nó consiste no número de nós destinos que esse nó pode alcançar, levando em consideração uma restrição topológica (*e.g.* distância ou número de saltos transparentes). Se houver nós com o mesmo grau lógico, será escolhido o nó que possuir maior grau físico como critério de desempate. Após o posicionamento de um nó regenerador, o grau lógico de todos os nós da rede será recalculado, para então verificar novamente o nó que possuir maior grau lógico e selecioná-lo. O processo é repetido até que todos pares de nós (origem, destino) da rede sejam alcançáveis.

Os algoritmos NDF e HNF são citados em [Nath *et al* 2014] como sendo algoritmos avançados e com bom desempenho para o posicionamento de nós regeneradores baseado em informações topológicas.

A métrica *betweennness* mede a centralidade dos nós da rede [Freitas 2010], de maneira que os nós da rede que são mais usados nos caminhos mínimos, são os nós que possuem maior grau de centralidade. Sendo assim, para a identificação do nó com maior grau de importância com base em sua centralidade, é calculada as rotas de menor caminho entre todos os pares de nós (origem, destino) da rede. Mediante todas as rotas apresentadas, a centralidade de um nó é obtida pela quantidade de menores caminhos que passam por ele. Neste trabalho, consideramos e avaliamos o uso da métrica *betweennnes* como uma estratégia de posicionamento de nós regeneradores denominada Centralidade dos Nós.

A estratégia baseada na métrica Distância Média utiliza a média do tamanho de todos os menores caminhos entre todos os pares de nós (origem, destino) da rede [Araújo *et al.* 2014], [Guimaraes *et al* 2015]. O menor caminho tratado neste trabalho é constituído pelo número mínimo de saltos em que o nó origem alcança o nó destino, calculado pelo algoritmo de menor caminho de Dijskstra.

A estratégia proposta chamada de IA - NDF consiste numa modificação no algoritmo NDF. Neste caso, será selecionado como nó regenerador aquele que possui o maior grau nodal (número de interfaces físicas), porém se houver mais de um nó com o mesmo grau físico essa escolha não é feita de forma aleatória como na versão original do NDF. Como critério de desempate o algoritmo IA - NDF escolhe o nó que possuir maior distância média dos enlaces diretamente ligados a ele.

O IA-NDF também proposto neste trabalho usa como critério de escolha dados da camada física. Segundo a literatura, os próprios equipamentos inserem as degradações de camada física [Soares *et al*. 2012], [Nath *et al.* 2014], portanto, quanto maior a distância do enlace, mais equipamentos serão utilizados e consequentemente haverá mais degradações nesse enlace. Sendo assim, ao incluir a distância como métrica para escolha do nó regenerador o algoritmo IA – NDF procura estabelecer um posicionamento permitindo mais qualidade no sinal da rede.

O segundo algoritmo proposto chamado de Modified HNF consiste numa adaptação do algoritmo HNF. Sendo assim, ele também seleciona como nó regenerador aquele que possui o maior grau lógico, e como critério de desempate será escolhido o nó que possuir maior grau físico. Contudo, após o posicionamento do regenerador no nó escolhido pelo algoritmo M-HNF proposto, o grau lógico dos nós não será recalculado como no HNF tradicional.

4. Resultados de Avaliação de Desempenho

Para os experimentos de simulação usamos a topologia Abilene e a Americana apresentada na Figura 1, os algoritmos de avaliados foram implementados pelos autores deste trabalho, e as simulações foram realizadas na ferramenta de simulação de Redes Ópticas WDM *Transparent Optical Network Simulator* (TONetS) [Soares *et al.* 2008]. Nas diversas simulações foram utilizadas os seguintes parâmetros: roteamento de menor caminho com o algoritmo de *Dijkstra*, para alocação de comprimento de onda, o algoritmo *FirstFit*, 4 replicações, 1.000.000 requisições de conexões ópticas, e 80 comprimentos de onda. Foram escolhidas as topologias Abilene (Figura 1a) [Duraes *et al* 2016] e a Americana (Figura 1b) [Duraes *et al* 2016].

Para os parâmetros de camada física foi admitido pelo menos 1 amplificador por link e no máximo 3 amplificadores na topologia Abilene, e no máximo 4 amplificadores da topologia Americana. Cada enlace da topologia Abilene variou de 80 a 240 km, enquanto na topologia Americana de 100 a 400 km. A diferença na configuração da distância e na colocação dos amplificadores ocorre devido ao tamanho real das topologias. Em ambas as topologias, a distribuição dos amplificadores foi feita de forma proporcional ao tamanho real (em km) da rede, sendo posto 1 a cada 80 km na topologia Abilene e 1 a cada 100 na topologia Americana. A sensibilidade do receptor utilizada foi de -25 dB, perda de inserção do *Switch* de 30 dB. Outros parâmetros de camada física foram os mesmos usados em [Fontinele *et al.* 2014] e [Zhao *et al.* 2014].

Para efeito de comparações, os gráficos apresentam também o desempenho da rede transparente (todos os nós transparente), e da rede opaca (todos os nós com regeneradores OEO).

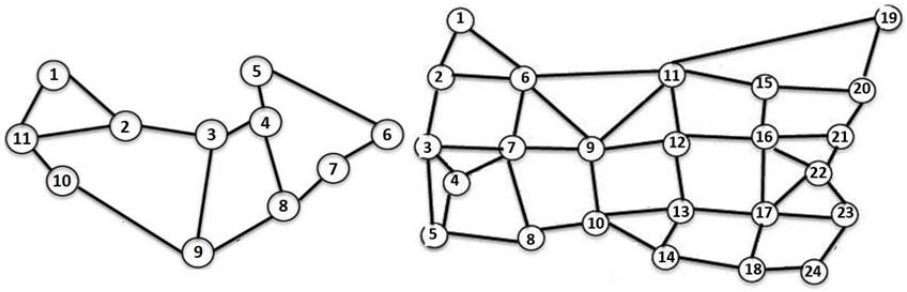


Figura 1 – Topologia da rede Abilene e Topologia da rede Americana [Duraes *et al.* 2016].

Na Tabela 1 é possível observar quais nós da rede foram selecionados para serem regeneradores na topologia Abilene segundo cada algoritmo simulado. Na Tabela 2, essas informações são dadas para a topologia Americana.

A probabilidade de Bloqueio Geral descreve em porcentagem quantas requisições foram bloqueadas. A Figura 2a apresenta a probabilidade de bloqueio dos algoritmos avaliados para a topologia Abilene e a Figura 2b para topologia Americana.

Tabela1 – Distribuição de regeneradores na topologia Abilene

Algoritmo	*Quantidade de Regeneradores*	*Nós Escolhidos*
NDF	6	2;4;9;6;11,7
IA – NDF	5	9;2;4;11;7
HNF	7	3;4;2;1;5;8;9
M – HNF	4	3;4;8;9
Centralidade dos Nós	5	3;4;2;9;8
Distância Média	9	1;10;7;11;6;9;5;8;2

Tabela2 – Distribuição de regeneradores na topologia Americana

Algoritmo	*Quantidade de Regeneradores*	*Nós Escolhidos*
NDF	12	11;7;16;17;10;6;3;20;12;22;5;14
IA – NDF	12	6;16;3;9;13;8;11;17;204;14;22
HNF	18	9;16;21;22;12;18;6;7;15;12;20;19;11;17;2;5;3;13
M – HNF	17	9;12;6;11;7;16;17;10;13;22;15;8;3;2;14;18;21
Centralidade	18	9;10;6;11;12;13;16;7;8;17;14;5;15;18;3;2;21;22
Distância Média	22	19;24;5;20;4;1;3;2;23;21;18;8;15;14;22;7;11;17;6 16;13;20

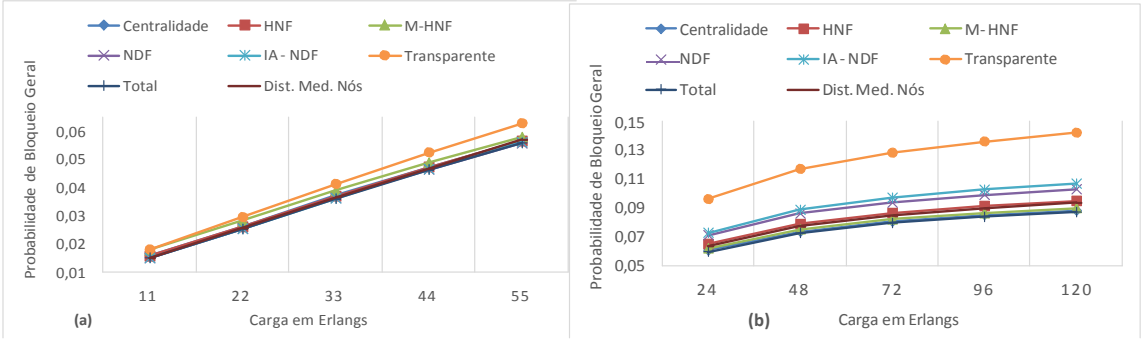


Figura 2 – Probabilidade de Bloqueio Geral

Na topologia Abilene, em geral, os algoritmos obtiveram um resultado bem semelhante, em termos de Probabilidade de Bloqueio (Figura 2a). Os algoritmos NDF e IA-NDF alcançaram resultados praticamente iguais e apresentaram os melhores desempenhos. Porém, é importante destacar que o algoritmo NDF posicionou um regenerador a mais que o algoritmo proposto IA-NDF, sendo assim, nesse cenário, o algoritmo proposto IA-NDF obteve um melhor resultado, ao serem analisadas as métricas de Probabilidade de Bloqueio e Quantidade de Regeneradores, em conjunto. Tal desempenho se justifica pelo fato do algoritmo proposto IA-NDF posicionar seus regeneradores em nós que favoreçam melhor a regeneração da qualidade do sinal óptico e, por consequência, a qualidade do sinal da rede será melhor e assim haverá menos bloqueio. Em termos de Quantidade de Regeneradores, o algoritmo proposto M-HNF apresentou melhor resultado ao posicionar apenas 4(quatro) nós regeneradores.

Na topologia Americana os algoritmos Centralidade e M-HNF apresentaram melhor desempenho, em termos de Probabilidade de Bloqueio (Figura 2b). Por outro lado, o algoritmo IA-NDF apresentou o pior desempenho em termos de Probabilidade de Bloqueio. O algoritmo Distância Média dos nós apresentou desempenho ligeiramente superior aos algoritmos M-HNF e o IA-NDF, porém, vale a pena destacar que ele posicionou uma quantidade de regeneradores muito maior que os outros, ao posicionar 22 nós regeneradores. O algoritmo proposto IA-DNF posicionou apenas 12 regeneradores, enquanto o M-HNF posicionou 17 regeneradores, nesta topologia. Desta forma, este último alcançou menor probabilidade de bloqueio na rede e o primeiro melhor desempenho em termos de quantidade de nós regeneradores posicionados. O algoritmo HNF utilizou 18 regeneradores, e, ainda assim, não obteve um bom desempenho em termos de probabilidade de bloqueio, quando comparado com o algoritmo de Centralidade e com o M-HNF, os quais também utilizaram 18 regeneradores.

Para se medir a degradação do sinal óptico, [Fontinele *et al.* 2016] apresenta como fator a relação sinal-ruído óptico (*Optical Signal Noise Ratio* - OSNR). Nesse caso, o OSNR deve estar em níveis aceitáveis para o sinal ser considerado de qualidade.

A Probabilidade de Bloqueio por OSNR é uma métrica que avalia quanto por cento das conexões foram bloqueadas somente por não satisfazer a qualidade do sinal óptico. Quando os algoritmos foram avaliados na topologia Abilene (Figura 3a), observou-se que os resultados foram semelhantes ao da probabilidade de bloqueio geral. Na topologia Abilene o IA-NDF obteve um resultado bem superior por sua característica de considerar a distância na escolha de enlaces com mesmo grau físico. O algoritmo distância média teve o pior desempenho, enquanto os outros algoritmos obtiveram resultados próximos.

Na topologia Americana os resultados da probabilidade de bloqueio por OSNR (Figura 3b) também ficaram semelhantes aos resultados da probabilidade de bloqueio geral, mas o algoritmo Centralidade apresentou ainda melhores resultados em relação aos outros algoritmos de posicionamento. O algoritmo M-HNF foi o segundo melhor, ficando o seu desempenho próximo ao desempenho do algoritmo de Centralidade, sendo que ele usou um regenerador a menos. Já o algoritmo IA-NDF, por dimensionar poucos regeneradores, não alcançou um bom desempenho, em termos de probabilidade de bloqueio, se comparado aos outros algoritmos.

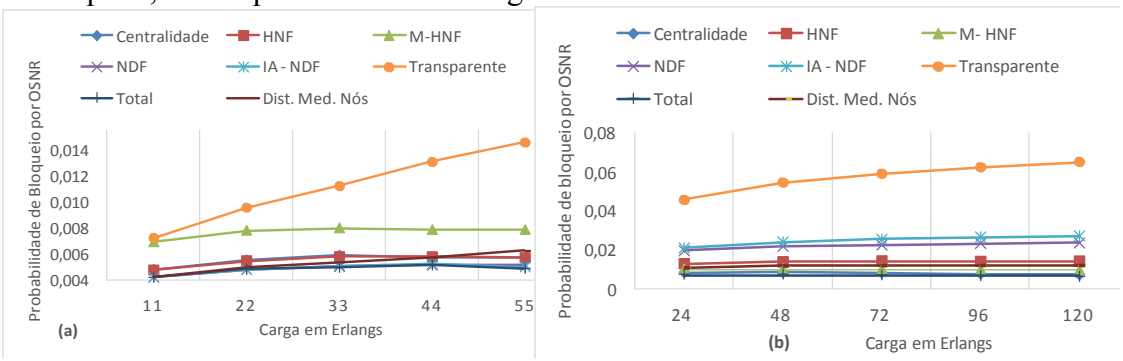


Figura 3 - Probabilidade de Bloqueio por OSNR.

5. Considerações Finais

O dimensionamento e posicionamento de nós regeneradores OEO é um grande desafio paras as redes ópticas translúcidas. Por este se tratar de um problema de otimização, a qualidade do sinal na rede varia de acordo com a heurística utilizada. Neste trabalho, foram avaliados algoritmos de dimensionamento e posicionamento de nós OEO apresentados na literatura que consideram informações da topologia da rede. Além disso, foram propostos e avaliados dois novos algoritmos, IA-NDF e M-HNF.

Ao avaliar o desempenho dos algoritmos de posicionamento na topologia Abilene, o algoritmo IA-NDF obteve melhor desempenho em termos de probabilidade de bloqueio, por sua característica de considerar a distância. Quanto à métrica de número de nós regeneradores posicionados, o algoritmo proposto M-HNF apresentou o melhor resultado para a topologia Abilene, ao posicionar ao menos um nó regenerador a menos que os demais algoritmos, sem comprometer a probabilidade de bloqueio da rede. Observou-se ainda diferentes níveis de desempenho dos algoritmos avaliados nas duas topologias, principalmente pela característica de uma topologia ser mais extensa que a outra, em termos de número de nós e enlaces.

Como trabalhos futuros sugere-se a inclusão de outras heurísticas baseadas em métricas espectrais [Freitas 2010], [Araujo *et al.* 2014], e a análise de experimentos em cenários de redes ópticas translúcidas elásticas [Fontinele *et al.* 2016].

6. Referências

Araújo, D. R. B., Bastos-Filho, C. J. A., Martins-Filho, J. F., (2014) "Métricas em Redes Complexas". Revista de Tecnologia da Informação, v. 4, n. 2, p. 11-18, Outubro, 2014.

Cisco (2015). "Cisco prevê o triplicação do tráfego IP entre 2014 e 2019", http://www.cisco.com/c/pt_pt/about/press/news-archive-2015/20150527.html, Maio.

Duraes, G. M., Araujo, V. V., Soares, A., Monteiro, J. A. S., Giozza, W. F. (2016) "An Iterative and Hybrid Strategy for Routing and OEO Placement in Translucent Optical Networks". In: Brazilian Symposium on Computer Networks and Distributed Systems (SBRC), p. 748-761, Maio 2016.

Fontinele, A., Santos, I., Durães, G. e Soares, A. (2016) "Achievement of fair and efficient regenerador allocations in translucent optical networks using the novel regenerador assignment algorithm". Optical Switching and Networking, Elsevier, 19, p. 22-39.

Fontinele, A., Santos, I., Durães, G., Maranhão, J., Soares, A (2014). Alocação Preventiva de Regeneradores em Redes Ópticas Translúcidas. In: 32 Simpósio Brasileiro de Computadores e Sistemas Distribuídos (SBRC). Florianópolis, Maio 2014.

Freitas, L. Q. (2010) "Medidas de Centralidade em Grafos". Dissertação (Mestrado em Engenharia de Produção) - Universidade Federal do Rio de Janeiro, Rio de Janeiro, 103f, 2010.

Guimaraes, A. C., Nascimento, A C. A. e Giozza, W. F. (2015) "Métricas de conectividade e vulnerabilidade em redes" Simpósio Brasileiro de Telecomunicações – (SBrT), Setembro 2015.

Nath, I., Chatterjee, M. e Bhattacharya, U. (2014) "A survey on regenerator Placement Problem in translucent optical network", in International Conference on Circuits, Systems, Communication and Information Technology Applications (CSCITA), p.408-413, 4-5, Abril.

Shen, G e Grover, W. D. (2004) "Segment-based approaches to survivable translucent network design under various ultra-long-haul system reach capabilities". Journal of Optical Networking, v. 3, n. 1, p. 1-24, Janeiro, 2004.

Soares, A. C. B. Moura, I. C.; Fontilene, A. C.; Durães, G. M.; Maranhão Neto, José C.; Assis, K. D. R (2012). "Redes Ópticas de Transporte: Transparentes vs. Translúcidas". In: Escola Regional de Computação do Ceará, Maranhão e Piauí (ERCEMAPI). (Org.). Livro de Minicursos da ERCEMAPI 2012. 1ed., v. 1, p. 1-24.

Soares, A. C. B., Durães, G. M., Giozza W. e Cunha, P. (2008) "TONetS: Ferramenta para Avaliação de Desempenho de Redes Ópticas Transparentes" in VII Salão de Ferramentas do Simpósio Brasileiro de Redes de Computadores - SBRC, Maio 2008.

Yang, X. e Ramamurthy, B. (2005). "Sparse Regeneration in Translucent Wavelength-Routed Optical Networks: Architecture, Network Design and Wavelength Routing", Photonic Network Communication, Volume 10, Number 1, July 2005, pp. 39-53.

Zhao, J., Subramaniam, S. e Brandt-Pearce, M. (2014) "Intradomain and Interdomain QoT-Aware RWA for Translucent Optical Networks". IEEE/OSA Journal of Optical Communications and Networking, v. 6. p. 536-548. Junho 2014.

II

ERRC - Sessão de Pôsteres

Fluxo de logs em ambiente de emulação CORE (Common Open Research Emulator)

Carlos de Moraes [1], Felipe Duarte[1], Luciano S da Silva[1]

[1]Centro de Processamento de Dados – Universidade Federal de Santa Maria (UFSM)
Caixa Postal 508 – 97105-900 – Santa Maria – RS – Brazil

{crgmoraes,luciano}@cpd.ufsm.br, felipe.dua@redes.ufsm.br

Abstract. *There are several tools that aid in the design and administration of computer networks. In this respect, the Common Open Research Emulator, also known as CORE, allows many tests of network operation before it is implemented as well as for network scalability tests or new services that are incorporated. However, because CORE uses a container for virtualization of applications running at the core of the system, this compromises the collection of activity logs of the node, since these are restricted to the host host and compromising the access of softwares That analyze the logs, since both use the same resource of the host.*

Resumo. *Existem diversas ferramentas que auxíliam tanto no projeto quanto na administração de redes de computadores. Neste aspecto o Common Open Research Emulator, também chamado de CORE, permite muitos testes de funcionamento de rede, antes mesmo desta ser implementada como também, para testes de escalabilidade da rede ou novos serviços que sejam incorporados. No entanto, pelo fato do CORE utilizar contêiner para virtualização de aplicações que executem no núcleo do sistema, isto compromete a coleta dos logs de atividades do nó, uma vez que estes ficam restritos ao host hospedeiro e comprometendo o acesso de softwares que analizam os logs, já que ambos utilizam o mesmo recurso do hospedeiro.*

1. Introdução

Para o projeto de uma rede que ainda está por ser implementada, é de considerável magnitude, que esta seja implementada em um simulador, bem como testada exaustivamente pois somente assim teremos que certeza que a rede estará longe de falhas por uma falha do projeto que foi descoberto após o seu projeto de execução. Para isto há diversas ferramentas de simulação de redes de computadores, entra elas o *CORE*, que é uma ferramenta que permite a implementação de uma rede virtual de computadores de maneira rápida através de uma interface customizável e de fácil utilização [CORE 2015]. O CORE fornece um ambiente para execução de aplicações e protocolos reais e pode ser conectado a roteadores e redes físicas existentes. Ele é usado para pesquisa de protocolos, demostrações, teste de plataformas, estudos de segurança, testes para aumento físico de rede, etc. Para a construção dos nós virtuais, o CORE utiliza o recurso de virtualização do Linux "network namespace" também conhecidos como nets ou Linux contêiners (LXC) junto com a construção de *Linux Ethernet bridging*.

2. Contêiner LXC

Contêiner é um recipiente que oferece um ambiente de execução o mais próximo possível de uma Máquina Virtual porém, sem a sobrecarga que acompanha a execução do núcleo separado e a simulação de todo o hardware do sistema computacional [LinuxContainers.org 2017]. Sistemas baseado em contêiners tem as aplicações que compartilham o mesmo sistema operacional resultando em aplicações que utilizam menos recursos. Cada conteiner deve fornecer aos seus processos a ilusão que não existem outros processos no sistema. Para isto o *namespace* deve fornecer uma visão isolada de um recurso global para o conjunto de processos participantes deste *namespace*. Porém tanto o núcleo do sistema operacional Linux quanto processos no espaço do usuário enviam seus registro de atividades para *rsyslogd* que classifica e grava em arquivos de registro "log" conforme a configuração utilizada [Kerrisk 2012].

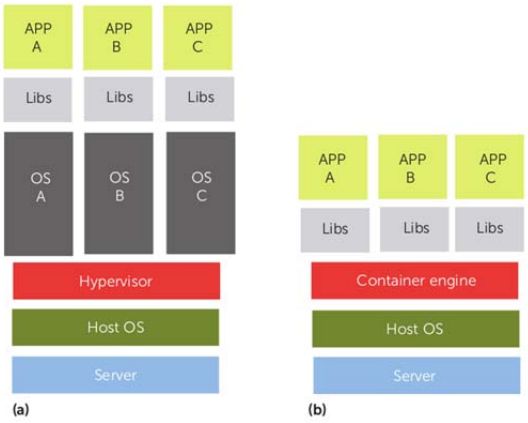


Figura 1. Comparação de sistemas: (a) *hipervisor* e (b) baseado em contêiner [Bernstein 2014].

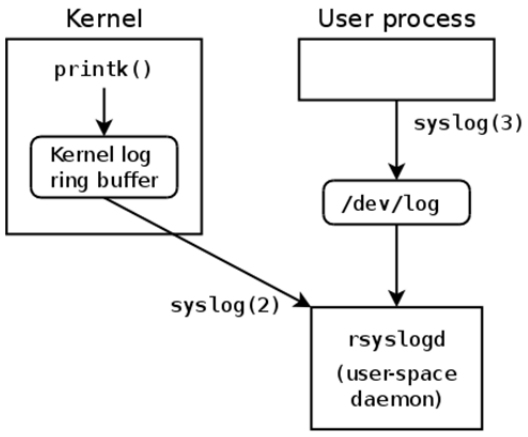


Figura 2. Fluxo de mensagens entre o núcleo e o *userspace* [Kerrisk 2012].

Análises realizadas em ambientes de redes dependem da visão dos arquivos de log gerados pelas aplicações. Em um acesso HTTP ao servidor web, normalmente é gerado

um log para esta atividade. O login ou a tentativa de login também deve ser registrado. Porém alguns registros partem do núcleo do Sistema operacional e não é disponibilizado ao nó cliente.

É possível a configuração de alguns serviços de modo que o registro ocorra dentro do espaço de usuário no emulador CORE como aplicação web mas não esta disponível para todos os serviços.

Netfilter é um framework de manipulação de pacotes de rede presente no Linux desde versões 2.4 do kernel do Linux ou posteriores. Normalmente é associado com o Iptables que é um interface de linha de comando que roda a partir do espaço do usuário permitindo a filtragem de pacotes, tradução de endereços/portas de rede, NA[P]T e outras modificações de pacotes [Netfilter 2017].

Aplicações de Firewall baseada no Netfilter executa apartir do núcleo no sistema operacional do hospedeiro enquanto que os comandos que são aplicados a partir do espaço do usuário como por exemplo o Iptables.

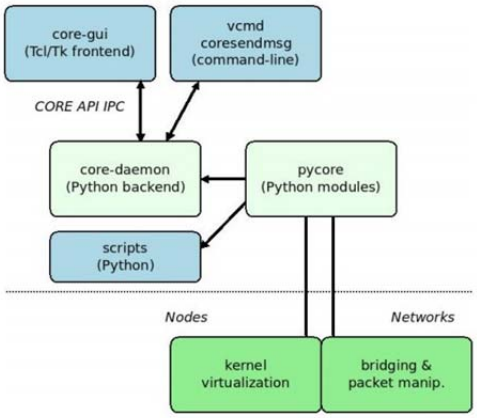


Figura 3. Arquitetura do CORE [CORE 2015].

De acordo com a figura 3, a linha tracejada representa a separação entre o espaço de usuário e o núcleo do sistema. A manipulação de pacotes é realizada abaixo desta linha pertencendo a área do núcleo.

3. Conclusão e Trabalhos Futuros

Este trabalho apresentou um ambiente de emulação de redes de computadores no qual se verificou que os registros de aplicações que são gerados pelo núcleo do sistema operacional não ficam disponibilizados para outras aplicações que estão dentro do *userspace* reservado para as aplicações de usuário, no caso o hosts emulados pelo CORE e suas aplicações locais sem acesso e não podem executar corretamente as suas funções. Uma proposta para o futuro pretende-se desenvolver uma aplicação que disponibilizasse estes registros para as aplicações que a requeressem.

Referências

Bernstein, D. (2014). Containers and cloud: From lxc to docker to kubernetes. In IEEE, editor, *IEEE Cloud Computing*, pages 81–84. IEEE.

CORE (2015). *CORE Documentation*. U.S. Naval Research Laboratory, 15th edition.

Kerrisk, M. (2012). *Stepping closer to practical containers: ”syslog”namespaces.* LWN.nety, 1th edition.

LinuxContainers.org (2017). Linuxcontainers.org infrastructure for container projects. Accesso em: Julho. 2017.

Netfilter (2017). The netfilter.org project. Accesso em: Julho. 2017.

Proposta de implementação de um balanceador de carga utilizando SDN e NFV

Gabriel Marchesan[1], Anderson M. da Rocha[1], Nilton C. B. da Silva[1], Ricardo B. Gomes[1], Roseclea D. Medina[1]

[1]GRECA - Grupo de Redes e Computação Aplicada
Programa de Pós-Graduação em Ciência da Computação
Universidade Federal de Santa Maria (UFSM) – Santa Maria, RS - Brasil

{gmarchesan,amonteiro,nbatista,ricardo,rose}@inf.ufsm.br

Abstract. *This work aims for the presentation a integration proposal of the technologies Software Defined Networks and Network Functions Virtualization through implementing a virtualized load balancer, solution called "NFV Load Balancer". Moreover, with this integration, can take advantage of the intelligence and global view of network of the SDN controller and gain better control of the instances implemented in NFV, obtaining many advantages provided by this feature by integrating these two new technologies area Computer Networks.*

Resumo. *Este trabalho tem por objetivo apresentar uma proposta de integração das tecnologias Software Defined Networking (SDN) e Network Function Virtualization (NFV) através da implementação de um balanceador de carga virtualizado, solução denominada de "NFV Load Balancer". Além disso, com esta integração, pode-se aproveitar a inteligência e a visão global da rede do controlador SDN e obter um melhor controle das instâncias implementadas em NFV, obtendo muitas vantagens proporcionadas por esta funcionalidade através da integração destas duas novas tecnologias da área de Redes de Computadores.*

1. Introdução

Apesar da grande expansão da Internet, em termos de quantidade de dados trafegados, de usuários conectados, de penetração e uma vasta gama de aplicações, não observou-se significativamente uma evolução em sua arquitetura nos últimos anos. Algumas modificações que já foram realizadas na arquitetura da Internet, não estão sendo suficientes para atender as demandas de novas aplicações que vem sendo inseridas todos dias na Rede.

Ao longo dos anos, a Internet tornou-se comercial, e os equipamentos de rede tornaram-se "caixas pretas", ou seja, implementações integradas verticalmente baseadas em *software* fechado sobre *hardware* proprietário [Matias *et al*., 2015]. Além disso, as redes vem se tornando parte da infraestrutura crítica de diversos ambientes, pois sua utilização é essencial em trabalhos empresariais, comerciais, domésticos, acadêmicos, entre outros. Com todos esses problemas surgindo, muitos pesquisadores afirmam que a arquitetura de redes de computadores em geral e a Rede Mundial de Computadores (Internet) atingiram um nível de amadurecimento que as tornaram pouco flexíveis

[Guedes *et al*., 2012]. O resultado desse modelo é o já reconhecido engessamento da Internet [Chowdhury and Boutaba, 2009].

Em contraste a essa realidade os pesquisadores de redes e a comunidade científica começaram a desenvolver propostas para a criação de novas tecnologias de redes de computadores, ou seja, novas arquiteturas de implementação do núcleo da rede. Nesse contexto, novas arquiteturas de redes são propostas, tais como, *Software Defined Networking* (SDN) [McKeown *et al.*, 2008] sendo traduzida para Redes Definidas por *Software* e mais recentemente a *Network Function Virtualization* (NFV) [ETSI, 2012], ou seja, a Virtualização das Funções de Rede.

Nesse sentido, este trabalho tem por objetivo apresentar uma proposta de integração destas duas tecnologias através da implementação de um balanceador de carga virtualizado, solução denominada de "NFV *Load Balancer*", mostrando como seria o seu funcionamento e os possíveis benefícios proporcionados por esta funcionalidade através da integração destas duas novas tecnologias de Rede.

A próxima seção apresenta os trabalhos relacionados. A seção 3, descreve a proposta de implementação do balanceador de carga utilizando SDN e NFV. Por fim, a seção 4 descreve as considerações finais deste trabalho.

2. Trabalhos Relacionados

Em [Liberato *et al.,* 2014] apresentou-se uma proposta de balanceamento de carga utilizando-se *hardware* de baixo custo e tecnologia SDN. A proposta de tais autores consiste em utilizar equipamentos não especializados de prateleira para demonstrar a viabilidade de habilitar o protocolo *OpenFlow* na versão 1.3, dessa forma permitindo a incorporação de novas características pelo projetista de rede. Para realizar tal projeto, foi realizada a substituição do sistema operacional *RouterOS,* que suporta apenas OF 1.0, pelo *OpenWRT* que permite a instalação de switches virtuais que suportam OF 1.3.

O trabalho de [Olaya, Bernal and Mejía, 2016] apresenta uma aplicação de balanceamento de carga para servidores *Web* utilizando os algoritmos de enfileiramento *First In, First Out* (FIFO), *Round Robin* (RR) e *Deficit Round Robin* (DRR). A aplicação foi desenvolvida para o controlador SDN Ryu, onde os testes foram realizados em uma rede real através de switches com suporte ao protocolo *OpenFlow* e em uma rede virtual utilizando o simulador Mininet. Para realizar os testes e verificar o correto funcionamento de cada algoritmo desenvolvido foi criado uma topologia onde alguns clientes realizavam requisições aos servidores *Web*. Além disso, verificou-se o comportamento e o tempo de resposta da aplicação realizando-se requisições utilizando-se os protocolos HTTP, HTTPS, tráfego de áudio e vídeo.

3. NFV *Load Balancer*

Embora SDN e NFV possam existir de forma independente, muito mais benefícios podem ser alcançados de forma eficaz quando estas duas tecnologias trabalham em conjunto. Através da integração destas duas tecnologias, pode-se aproveitar a inteligência de um controlador SDN para gerir funcionalidades implementadas em NFV de acordo com o estado e as demandas da rede. Além disso, novas instâncias virtualizadas de rede podem ser inicializadas conforme as mudanças e necessidades das aplicações e de negócios do cliente [Han *et al.,* 2015].

O serviço de balanceamento de carga é muito importante nas redes de computadores, tem vantagens e características, tais como: aumentar a escalabilidade e a flexibilidade da rede quando é adicionado e requisitado o processamento de novos recursos; aumentar o desempenho da rede pela utilização dos recursos de forma equilibrada, transparente aos usuários finais; tolerância a falhas, já que se um servidor falhar a carga pode ser redistribuída aos outros servidores existentes sem comprometer o processamento das requisições, etc.

O balanceador de carga virtualizado proposto nesta pesquisa utiliza em conjunto as tecnologias SDN e NFV. Nesse sentido, além de ter um custo menor, tem algumas características particulares e oferece muitas vantagens em relação ao balanceador de carga tradicional implementado em *hardware* dedicado com tecnologia proprietária. Utilizando-se está função de rede virtualizada, as regras de balanceamento de carga podem ser definidas de acordo com as várias aplicações existentes, e o equilíbrio de carga pode ser feito com a visão global da rede que o controlador SDN possui. Nesse contexto, por exemplo, se houver uma falha no caminho da rede em particular, as regras de fluxos podem ser alteradas de modo que um caminho diferente é estabelecido para alcançar a entidade da VNF, além disso, a largura de banda e a qualidade de serviço também podem ser ajustadas dinamicamente para determinados fluxos que deseja-se ter prioridade no tráfego da rede.

Ainda, se a demanda por capacidade de balanceamento de carga aumenta, a camada de orquestração de rede pode rapidamente alocar novas instâncias de balanceamento de carga e também ajustar a rede de comutação de infraestrutura para acomodar os padrões de tráfego alterados. Por sua vez, a entidade VNF de balanceamento de carga pode interagir com o controlador SDN para avaliar o desempenho e a capacidade da rede em determinados intervalos de tempo e usar estas informações adicionais para equilibrar melhor o tráfego entre as entidades responsáveis pelo balanceamento de carga. Além do mais, pode-se até mesmo solicitar o aprovisionamento de novas instâncias virtuais adicionais para melhorar a realização deste serviço ou então liberar estas instâncias caso as mesmas não sejam mais necessárias, desta forma liberando recursos computacionais e também provendo economia energética. A Figura 1 ilustra, de forma simplificada, o funcionamento do balanceador de carga proposto neste trabalho.

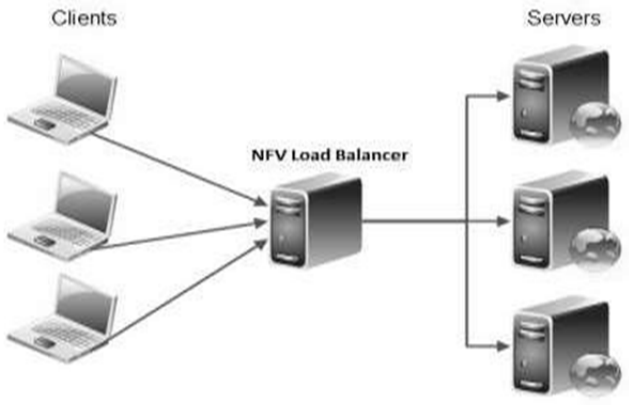


Figura 1 – NFV Load Balancer

Fonte: Autoria própria

4. Considerações Finais

Através desta proposta de integração das tecnologias SDN e NFV pode-se implementar em *software* a funcionalidade de balanceamento de carga, sendo a mesma virtualizada em *hardware* de uso geral, não necessitando de gastos significativos com o uso de *hardware* especializado (*middlebox*) com tecnologia proprietária, logo diminuindo o custo do balanceador de carga. É importante ressaltar que é possível implementar as tecnologias SDN e NFV de forma separada, mas os benefícios proporcionados por ambas as tecnologias podem ser mais eficazes quando as mesmas são utilizadas em conjunto. Desta forma, pode-se aproveitar a inteligência e a visão global da rede do controlador SDN e obter um melhor controle das instâncias implementadas em NFV de acordo com o estado e demandas da rede.

Além disso, novas instâncias virtualizadas de rede podem ser inicializadas conforme as mudanças e necessidades das aplicações e de negócios do cliente. Ainda, se a demanda por capacidade de balanceamento de carga aumenta, através da comunicação do controlador SDN com a entidade VNF, pode-se aprovisionar novas instâncias virtuais ou então liberar instâncias virtuais que não sejam mais necessárias para realizar o serviço em questão. Desta forma, é possível maximizar os recursos computacionais e também diminuir os gastos energéticos.

Referências

Chowdhury, N.; Boutaba, R. (2009) "Network virtualization: state of the art and research challenges" em IEEE Communications Magazine, v.47, n.7, p.20–26.

Guedes, D. *et al.* (2012) "Redes Definidas por Software: uma abordagem sistêmica para o desenvolvimento das pesquisas em redes de computadores". XXX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC, p.160–210.

Han *et al.* (2015) "Network Functions Virtualization: Challenges and Opportunities for Innovations", em IEEE Communications Magazine, v.53, n.2, p.90-97.

Liberato, A.; Mafioletti, D.; Spalla, E.; Martinello, M.; Villaça, R. (2014) "Balanceamento de Carga em SDN Provido por Comutadores Estocásticos de Prateleira", em Anais do V Workshop de Pesquisa Experimental da Internet do Futuro - WPEIF 2014, p.13-16.

Matias, J., Garay, J., Toledo, N., Unzilla, J., and Jacob, E. (2015). "Toward an sdn-enabled nfv architecture". IEEE Communications Magazine, 53(4):187–193.

McKeown, N. *et al.* (2008) "OpenFlow: enabling innovation in campus networks", em ACM SIGCOMM Computer Communication, p.1–6.

NFV White Paper. (2012) "Network Functions Virtualisation: An Introduction, Benefits, Enablers, Challenges & Call for Action", Disponível em: http://portal.etsi.org/NFV/NFV_White_Paper.pdf. Acesso em: Julho, 2017.

Olaya, M. E.; Bernal, I; Mejía, D. (2016) "Application for load balancing in SDN," *em 8th Euro American Conference on Telematics and Information Systems (EATIS)*, Cartagena, 2016, p. 1-8.

Campus Test Cloud: Uma Proposta de Testbed Universitária

Ricardo Bianchin Gomes[1], Gabriel Marchesan[1], Anderson M. da Rocha[1], Nilton C. Batista[1], Roseclea D. Medina[1]

[1]Programa de Pós-graduação em Ciência da Computação – Universidade Federal de Santa Maria (UFSM)

{ricardo, gmarchesan, amonteiro, nbatista, rose}@inf.ufsm.br

Abstract. *The need of large-scale computing research have stimulating the creation of testbeds. At the same time, computing resources virtualization through cloud computing have been showing itself as a very attractive tool for institutions, by providing savings, good performance and better availability of services. However, access to these technologies is not well diffused in public education institutions. So, this works proposes an architecture of computing that integrates cloud computing and testbeds inside universities, the Campus Test Cloud.*

Resumo. *A necessidade de realização de pesquisas de computação em larga escala vem estimulando a criação de redes de experimentação. Ao mesmo tempo, a virtualização de recursos computacionais através das clouds vem se demonstrando uma ferramenta muito atraente para as instituições, por prover economia, bom desempenho e melhor disponibilidade dos serviços. Contudo, o acesso a ambas as tecnologias ainda não é bem difundido nas instituições públicas de ensino. Assim, este trabalho propõe uma arquitetura de computação que integra cloud computing e testbeds dentro das universidades, a Campus Test Cloud.*

1. Introdução

A Internet cresceu muito nas três últimas décadas, e tem sido bem-sucedida em lidar com aplicações distribuídas. Contudo, apesar do crescimento, a Internet ainda se utiliza da mesma arquitetura inicial, sofrendo com as limitações do modelo TCP/IP. Ainda, a sua disseminação e popularidade acabam por ser outro fator limitante para a continuidade de sua expansão, uma vez que a adoção de novos padrões de funcionamento devem atender a diversos interesses [Chowdhury e Boutaba 2010]. Desta forma, muitos autores acreditam que uma solução para este impasse seria um redesenho da atual arquitetura da Internet [Farias *et al.* 2011].

No intuito de viabilizar pesquisas de novos protocolos e tecnologias que permitam o avanço da rede mundial, surgiram as redes de experimentação, também conhecidas como *testbeds*. Estas redes consistem de um ambiente de computação onde é possível executar testes em larga escala e desenvolver novos protocolos e/ou novos serviços em Redes de Computadores (RC), embora seu uso não esteja limitado a esta área. Graças aos avanços de novas tecnologias de Redes Definidas por *Software* (SDNs) e virtualização de redes, as *testbeds* podem ser virtualizadas, operando em conjunto com uma rede de produção,

reduzindo custos de implantação e manutenção. Portanto, virtualização pode acelerar o desenvolvimento de novos conceitos em RC [Blenk *et al*. 2015].

Porém, mesmo com o crescimento destas tecnologias, grupos individuais de pesquisa em universidades públicas brasileiras muitas vezes carecem de recursos computacionais para o desenvolvimento de seus projetos. Assim, este trabalho vem propor uma arquitetura modelo de *testbed* universitária para ser implantada em campi de universidades, de forma a potencializar a realização de pesquisas em computação.

2. Proposta: Campus Test Cloud (CTC)

O principal objetivo da arquitetura CTC é prover uma rede de experimentação, fomentando a pesquisa, juntamente com a rede de produção da instituição, de forma a maximizar o aproveitamento de *hardware* e consumo de energia. A arquitetura pode ser vista como uma nuvem universitária, distribuída entre diversos centros de ensino, com infraestrutura de rede baseada em SDN (*OpenFlow*). Esta nuvem provê serviços de terminal para os usuários da instituição e recursos virtuais de computação e de redes para a execução de experimentos de pesquisa, sendo baseada em tecnologias *open source* como o OpenStack ou OpenNebula.

As principais vantagens em se empregar esta arquitetura consistem em: 1) aproximação dos pesquisadores às tecnologias de *testbeds*, fomentando o desenvolvimento de pesquisas acadêmicas; 2) gerenciamento centralizado de recursos; 3) melhor disponibilidade de serviços e 4) economia de energia, graças ao melhor aproveitamento de recursos de *hardware*.

A figura 1 apresenta a organização geral do modelo proposto. A arquitetura é composta de três principais componentes: Centro de Processamento de Dados (CPD), Núcleos Prediais e Clientes de Terminal, que serão apresentados a seguir.

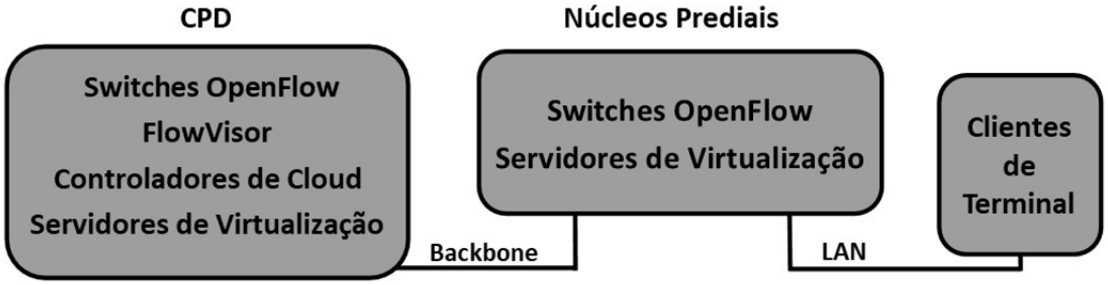


Figura 1. Arquitetura da Campus Test Cloud

2.1 Centro de Processamento de Dados

No CPD, encontram-se *switches OpenFlow*, responsáveis por comutar o tráfego de ambas as redes de experimentação e de produção do campus. Ambos os tipos de tráfego podem coexistir simultaneamente dentro da mesma infraestrutura física, graças ao "fatiamento" dos recursos de rede, que é provido pelo *hypervisor* de rede *FlowVisor* (FV). Para isso, existe um servidor dedicado à execução deste controlador SDN especializado. Cada *slice* criado pelo FV possui seu próprio controlador SDN dedicado em execução.

O CPD possui ainda outros dois componentes: os Controladores de *Cloud* (CC) e os Servidores de Virtualização (SV). Os CCs são responsáveis por orquestrar a utilização dos recursos de *cloud computing*, distribuídos entre o campus. Os SV provêm recursos

virtualizados para a rede de experimentação e a execução de máquinas virtuais e/ou containers para a nuvem privada da instituição.

A alocação de recursos para experimentação se dá através de um *framework* de gerenciamento e controle, serviço também fornecido pelo CPD. Através dele, um usuário pode criar um experimento dentro da *testbed* universitária, alocar recursos e executá-lo, de forma similar à encontrada nas grandes *testbeds*.

2.2 Núcleos Prediais

Os Núcleos Prediais (NPs) podem ser vistos como CPDs menores, ou como "ilhas", localizados em cada prédio da instituição. Suas principais funções são: 1) disponibilizar recursos computacionais e de rede para a *testbed* e 2) virtualizar as estações de trabalho e prover serviços com baixa latência aos usuários de terminal. Eles são compostos de ao menos um *switch OpenFlow* e um servidor de virtualização.

Os NP são ligados ao CPD através de um *backbone* redundante de fibra óptica, pelo qual trafegam dados de produção e de experimentação. Os *switches OpenFlow* de cada NP são pertencentes à infraestrutura SDN como um todo, sendo assim gerenciados pelos controladores existentes no CPD.

Os servidores de virtualização existentes em cada NP são responsáveis por virtualizar as estações de trabalho dos usuários e prover recursos computacionais para a *testbed*. A virtualização de estações de trabalho é um ponto vantajoso, pois através desta técnica, obtém-se redução no consumo total de energia, melhor aproveitamento dos recursos computacionais e economia de espaço físico [Shikida *et al.* 2012]. Os SV são componentes integrantes à *cloud* privada como um todo, sendo que seu gerenciamento é feito essencialmente pelo CC existente no CPD.

2.3 Clientes de Terminal

Os Clientes de Terminal são as estações de trabalho finais dos usuários que se encontram dentro da instituição. Constituem-se de *thin clients* conectados pela rede predial até o NP. As principais vantagens em se usar *thin clients* no lugar de computadores convencionais são o baixo custo de aquisição e baixo consumo de energia.

Ao ligar o terminal, o cliente pode escolher o Sistema Operacional (SO) desejado através de uma interface gráfica. No momento em que o SO for escolhido, o controlador da nuvem solicita a um servidor de virtualização do núcleo predial em questão para que crie uma instância de máquina virtual para ser utilizada pelo cliente.

3. Trabalhos Relacionados

Shikida *et al.* (2012) reportam um caso de sucesso em implantação de uma nuvem privada em uma universidade no Japão, utilizando-se *VMware vSphere* para a virtualização de *hosts*. A função desta nuvem é prover armazenamento de dados e serviços de terminais aos alunos e funcionários da instituição. Os principais resultados alcançados através desta arquitetura foram: melhor utilização de recursos computacionais; economia de 48% de energia elétrica e economia de 70% de espaço. Porém, a arquitetura apresentada não são disponibiliza serviços de *testbed* aos usuários.

Outras *testbeds* tem sido desenvolvidas recentemente, como GENI, OFELIA, FIBRE. A primeira visa desenvolvimento e validação de tecnologias de redes de

computadores na américa do norte [Berman *et al.* 2014]. Já a segunda é uma *testbed* europeia baseada em OpenFlow, composta de *testbeds* individuais providas por entidades parceiras do projeto [Suñé *et al.* 2014]. Por fim, a FIBRE é uma rede de experimentação que visa promover pesquisas em Internet do futuro no Brasil [Salmito *et al.* 2014].

Enquanto as *testbeds* acima apresentadas visam apenas fornecer recursos para experimentação, este trabalho propõe o compartilhamento de infraestruturas físicas para ambos fins de experimentação e produção, nas instituições de ensino, maximizando o aproveitamento do *hardware* instalado.

4. Considerações Finais

A necessidade de se desenvolverem pesquisas de novas tecnologias em redes de computadores e em outras áreas da computação foram fatores determinantes para o surgimento das *testbeds*. Contudo o acesso a redes de experimentação e recursos computacionais para a realização de experimentos ainda se demonstra insuficiente dentro de instituições públicas de ensino no Brasil.

No intuito de alavancar a pesquisa em informática e em áreas afins, apresentou-se, neste trabalho, uma proposta de arquitetura que converge tecnologias SDN e *cloud computing*, de forma a criar uma *testbed* universitária operando em conjunto com uma nuvem privada. Assim, almeja-se que o modelo proposto consiga: 1) estimular o desenvolvimento de pesquisas, por aproximar os usuários a um ambiente de experimentação real; 2) prover economia de energia, através da virtualização de estações de trabalho e migração destas para a nuvem privada e 3) prover melhor disponibilidade dos serviços, assim como facilidade de seu gerenciamento e 4) redução de custos de infraestrutura, pois esta é compartilhada.

Referências

Berman, M. *et al.* (2014) "GENI: A federated testbed for innovative network experiments", em Computer Networks, vol. 61, p. 5-23;

Blenk, A., Basta, A., Reisslein, M. Kellerer, W. (2015) "Survey on network virtualization hypervisors for Software Defined Networking".

Chowdhury, N. M. M. K. e Boutaba, R. (2010) "A survey of network virtualization", em Computer Networks, vol. 54, ed. 5, p. 862-876.

Farias *et al.* (2011) "Pesquisa experimental para a Internet do futuro: uma proposta utilizando virtualização e o framework OpenFlow", em XXIX Simpósio de Redes de Computadores e Sistemas Distribuídos - SBRC

Guedes, D. *et al*. (2012) "Redes definidas por software: uma abordagem sistêmica para o desenvolvimento das pesquisas em Redes de Computadores".

Salmito, T. *et al.* (2014) "FIBRE – An International Testbed for Future Internet Experimentation", em Anais do 32° SBRC, 2014.

Shikida, M. *et al.* (2012) "An Evaluation of Private Cloud System for Desktop Environments", em 40th Annual ACM SIGUCCS Conference on User Services, p. 131-134.

Suñhé, M. *et al.* (2014) "Design and Implementation of the OFELIA FP7 Facility: The European OpenFlow Testbed", em Computer Networks, vol 61, p.132-150.

Uma proposta para o monitoramento energético de nuvens computacionais privadas no Zabbix

Raul Leiria[1], Adriano Vogel[1], Dalvan Griebler[1], Claudio Schepke[2]

[1]Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Porto Alegre – RS – Brasil

[2]Universidade Federal do Pampa (UNIPAMPA)
Alegrete – RS – Brasil

{raul.leiria,adriano.vogel,dalvan.griebler}@acad.pucrs.br

claudioschepke@unipampa.edu.br

Resumo. *Computação em Nuvem se consolidou nos últimos anos como um novo paradigma computacional devido a sua ampla utilização. Isso resulta em um aumento do consumo de energia nos data centers. Consequentemente, existe uma crescente demanda por monitoramento energético em infraestruturas computacionais. Dessa forma, nesse trabalho é proposto um mecanismo para monitoramento do consumo energético para infraestruturas de nuvem usando a ferramenta de código aberto Zabbix.*

Abstract. *In the last years, cloud computing has been consolidated as a new computational paradigm due to its widespread adoption. Proportionally, this leverages to the increasing of the power consumption by data centers. As a consequence, there is a witnessed about the growing demand for monitoring the power consumption on computational infrastructures. Therefore, in this work is proposed a mechanism to monitor the cloud computing power draw through the Zabbix open-source networking monitoring tool.*

1. Introdução

A computação em nuvem é um paradigma que vem sendo amplamente adotado nos últimos anos por usuários domésticos e empresariais. A facilidade do sistema *Pay-As-You-Go* [Aldossary and Djemame 2016] permite ao usuário alocar recursos computacionais conforme às necessidades de suas aplicações e pagar apenas pelo que se utiliza. Além disso, a computação em nuvem oferece diferentes níveis de interação para o usuário [Vogel et al. 2016], sendo possível contratar desde máquinas virtuais até serviços de mais alto nível como *object storage* e *Content Delivery Networks* (CDNs).

Nuvens computacionais podem ser implantadas tanto em *data centers* de terceiros quanto em *data centers* locais pertencentes ao usuário. No primeiro caso trata-se de uma nuvem pública [Buyya et al. 2013], onde a infraestrutura para hospedagem é alugada, podendo o usuário escolher quais níveis de interação são convenientes para gerenciar e hospedar suas aplicações. No segundo caso trata-se de uma nuvem privada [Buyya et al. 2013], onde o usuário é responsável pela gerência de toda sua infraestrutura. Nesse caso, a latência para serviços locais é bastante baixa em comparação à nuvem pública e o usuário possui maior liberdade em todos os níveis que compõem esse sistema computacional.

2. Motivação

Nuvens privadas são orquestradas por ferramentas gerenciadoras e específicas para esse fim. Essas ferramentas são modularizadas para atender diferentes tipos de demandas computacionais como virtualização, armazenamento de bloco e de objetos, rede, alta disponibilidade e confiabilidade. Usualmente, gerenciadores de nuvem permitem interação via *Command Line Interface* (CLI), *Application Programming Interface* (API) e interface *Web*. As próprias plataformas de nuvem disponibilizam diferentes tipos de informações e métricas acerca de seu estado, sendo ainda possível a integração dessas métricas com ferramentas para monitoramento de ativos de rede, como o Zabbix [Dalle Vacche 2015] que se comunica através do protocolo *Simple Network Management Protocol* (SNMP) [Mauro and Schmidt 2005].

Apesar disso, as atuais ferramentas para gerenciamento de nuvem [Leiria et al. 2016] e monitoramento de ativos de rede não possuem mecanismos nativos para estimar ou medir o consumo de energia elétrica dos recursos computacionais em uso. Há uma grande preocupação por parte da indústria e da academia em relação ao consumo de energia elétrica dos *data centers*, uma vez que eles são responsáveis por a emissão de pelo menos 2% do dióxido de carbono mundial [Schulz 2009]. Por essas razões, neste trabalho é proposto um mecanismo para o monitoramento energético de nuvens privadas de forma integrada com a ferramenta Zabbix.

3. Trabalhos relacionados

Com a crescente adoção de nuvens privadas por provedores e empresas, monitorá-las tem se tornado crucial. Em vista disso, [Skvortsov et al. 2016] realizaram uma análise comparativa de dois *frameworks* europeus intitulados respectivamente ECO2Clouds e EXCESS. Ambos monitoram o consumo energético de nuvens por meio do Zabbix com o propósito de reduzir a emissão de CO2. Foi concluído que o EXCESS possui maior precisão a nível de servidor físico ao passo que o ECO2Clouds se diferencia a nível de virtualização e aplicações.

No trabalho de [Aldossary and Djemame 2016] os autores observaram que os atuais provedores de nuvem geralmente cobram o usuário por tempo de utilização dos recursos independente da quantidade de uso e da energia consumida. Por esse motivo os autores implementaram um algoritmo intitulado *Cost and Energy Aware Scheduling* (CEAS) para relacionar preço, energia elétrica consumida e desempenho. Testes foram realizados em um Testbed baseado em OpenNebula onde suas instâncias foram monitoradas utilizando o Zabbix. Quando utilizado o algoritmo dos autores foi possível obter até 63,3% de redução nos gastos monetários totais com energia elétrica em relação aos atuais modelos de cobrança.

[Iqbal et al. 2016] afirmam que vários trabalhos estão focados apenas no consumo energético agregado (total) dos *data centers* de nuvem, havendo por isso possibilidades de otimização em formas granulares, como a nível de Unidade de Processamento Central (CPU). Logo, nesse trabalho os autores propuseram monitorar os processadores dos servidores com a ferramenta Zabbix via protocolo SNMP. O objetivo é determinar o consumo de energia de cada servidor em razão de seus níveis de uso das unidades de processamento.

Apesar dos trabalhos apresentados nesta seção utilizarem o Zabbix para realizar o monitoramento energético de nuvens computacionais, nenhum deles utiliza os recursos de *triggers* e alarmes para emitir avisos sobre eventuais picos de consumo de energia elétrica que possam inclusive levar à violação de *Service Level Agreement* (SLAs) junto à concessionária de fornecimento de energia. Além disso, as *triggers* do Zabbix podem ser utilizadas para que ações sejam executadas sempre algum limite pré-definido for atingido.

4. Arquitetura

Nesta primeira etapa do trabalho será modelado o mecanismo para realizar o monitoramento energético em nuvens computacionais de maneira que seja possível a integração com o Zabbix, conforme pode ser visto na Figura 1. Em cada servidor de computação da nuvem haverá um módulo designado para listar quais instâncias estão sendo executadas. No sistema operacional do servidor hospedeiro, cada instância de máquina virtual é representada por um identificador de processo (PID) [Tanenbaum and Herbert 2014] que permite obter estatísticas acerca de quais recursos computacionais estão sendo utilizados junto às respectivas quantidades de uso.

Apesar dos mecanismos utilizados para o monitoramento de energia estarem disponíveis em *hardware*, é possível utilizar modelos matemáticos para estimar o consumo de energia elétrica de cada processo do sistema operacional. Em geral, os modelos de energia recebem estatísticas de uso do sistema operacional com a porcentagem respectiva a cada identificador de processo das instâncias em conjunto com dados energéticos obtidos através de sensores do *hardware*. Essas informações são aplicadas como entradas no modelo matemático para obter à estimativa energética de cada instância de máquina virtual.

O mecanismo proposto será agnóstico em relação ao sistema operacional de instâncias de máquinas virtuais e de suas aplicações internas em execução. Apesar disso, o determinante da quantidade de energia elétrica consumida pela máquina virtual são as cargas de trabalho executadas por suas aplicações. Todavia, mesmo que se desconheça quais aplicações estão em execução o monitoramento energético das instâncias de máquinas virtuais poderá ser realizado, visto que a sua execução sempre ocorre em nível de servidor hospedeiro.

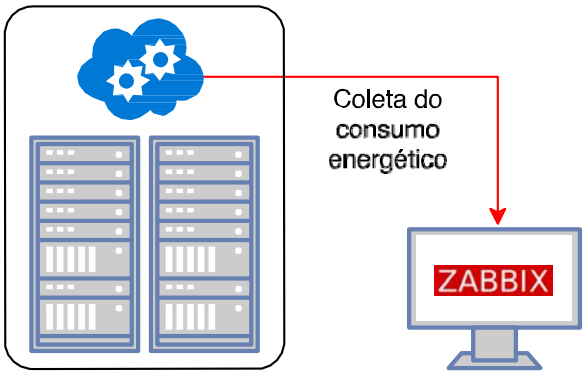


Figura 1. Arquitetura do mecanismo proposto

5. Resultados esperados

Nas próximas etapas será realizado o desenvolvimento do modelo proposto neste trabalho. Se espera que em situações onde houver picos no consumo de energia elétrica seja possível registrar no banco de dados do Zabbix quais componentes da nuvem estão levando ao maior consumo, com isso levando à emissão de alertas sobre o estado energético em que a nuvem se encontra. Em um segundo momento se espera com base nas informações das métricas de energia ser possível tomar decisões de modo que instâncias possam ser desligadas ou realocadas em servidores físicos que estejam mais propícios a hospedá-las.

Referências

Aldossary, M. and Djemame, K. (2016). Energy Consumption-based Pricing Model for Cloud Computing. In *32nd UK Performance Engineering Workshop*, pages 16–27. University of Bradford.

Buyya, R., Vecchiola, C., and Selvi, S. T. (2013). *Mastering cloud computing: foundations and applications programming*. Newnes.

Dalle Vacche, A. (2015). *Mastering Zabbix*. Packt Publishing Ltd.

Iqbal, A., Pattinson, C., and Kor, A.-L. (2016). Managing energy efficiency in the cloud computing environment using SNMPV3: A quantitative analysis of processing and power usage. In *Dependable, Autonomic and Secure Computing, 14th Intl Conf on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech), 2016 IEEE 14th Intl C*, pages 239–244. IEEE.

Leiria, R., Schepke, C., Mello, A., and Griebler, D. (2016). Um monitor de consumo energético para computação em nuvem na ferramenta OpenNebula. *Simpósio de Sistemas Computacionais de Alto Desempenho (WSCAD)*.

Mauro, D. and Schmidt, K. (2005). *Essential SNMP: Help for System and Network Administrators*. ”OŔeilly Media, Inc”.

Schulz, G. (2009). *The Green and Virtual Data Center*.

Skvortsov, P., Hoppe, D., Tenschert, A., and Gienger, M. (2016). Monitoring in the Clouds: Comparison of ECO2Clouds and EXCESS Monitoring Approaches. *arXiv preprint arXiv:1601.07355*.

Tanenbaum, A. and Herbert, B. (2014). *Modern Operating Systems*. ”Pearson”.

Vogel, A., Griebler, D., Maron, C. A., Schepke, C., and Fernandes, L. G. (2016). Private IaaS clouds: a comparative analysis of OpenNebula, CloudStack and OpenStack. In *Parallel, Distributed, and Network-Based Processing (PDP), 2016 24th Euromicro International Conference on*, pages 672–679. IEEE.

UbiPri PRIPRO - Controle e Gerenciamento de Perfis de Usuários com Base na Privacidade de Dados

Douglas Almeida dos Santos, Jonas Cesconetto, João A. Martins, Luis A. Silva, Iago S. Ochôa, Valderi R.Q. Leithardt

Laboratório de Sistemas Embarcados e Distribuídos – LEDS
Universidade do Vale do Itajaí (UNIVALI)
Caixa Postal 360 – CEP 88302-202 – Itajaí, SC – Brasil

{douglasas, jonascesconetto, joao_martins, luis.silva}@edu.univali.br, isochoa95@gmail.com, valderi@univali.br

Abstract. *This article presents the development of a module to the UbiPri middleware that manages the user profiles in ubiquitous environments. The main contribution is making the environments safer through the delimitation of resources and services to each profile.*

Resumo. *Este artigo apresenta o desenvolvimento de um módulo para o middleware UbiPri que gerencia os perfis de usuários em ambientes ubíquos. A sua principal contribuição é tornar os ambientes mais seguros por meio da delimitação de recursos e serviços para cada perfil.*

1. Introdução

A Internet das Coisas é uma rede interconectada de objetos inteligentes que se comunicam sem interação humana [Thakare et al. 2016]. Esta é uma área emergente na qual estima-se bilhões de dispositivos conectados à Internet em 2020 a qualquer momento e lugar, apresentando sérios desafios de segurança e privacidade, estes que estão entre as maiores barreiras para o desenvolvimento de Internet das Coisas em grande escala [Satyanarayanan 2017] [Cherkaoui et al. 2014].

A comunicação entre os dispositivos ocorre de maneira ubíqua, tornando-os uma parte do ambiente e facilitando a vida das pessoas em ambientes ubíquos por meio da coleta e processamento de informações sem que haja intervenção humana. Porém, cuidados devem ser observados com o manuseio da informação e perfil individual, buscando garantir segurança e privacidade [Leithardt et al. 2013]. Para que a informação seja trocada de maneira eficiente, deve ser garantida a compatibilidade entre os dispositivos, um dos meios de prover isso é a utilização de serviços web.

Neste contexto, foi desenvolvido o módulo de controle de perfis (PRIPRO - *PRIvacy PROfiles*) para o middleware UbiPri (*Ubiquitous Privacy*) desenvolvido por [Leithardt et al. 2013], que utiliza serviços web com o modelo arquitetural REST (*REpresentational State Transfer*). Esse middleware é responsável por definir, para cada usuário, limitações no uso de objetos inteligentes presentes nos ambientes. Neste estudo os objetos inteligentes são nomeados de recursos e as tecnologias como WiFi, GPS, Bluetooth, nomeadas de serviços, utilizando perfil individual.

2. Estado da Arte

[Corrad et al. 2004] desenvolveu um middleware chamado UbiCOSM *Security Framework* que controla o provisionamento de recursos com base na disponibilidade do recurso e na identidade/papel do usuário. Além disso, ele gerencia a visibilidade que o usuário tem dos recursos e de outros usuários presentes no ambiente dependendo do seu nível hierárquico. Porém, diferentemente deste artigo, não parte do princípio que o perfil dos usuários é definido para cada ambiente.

[Nunes 2013] desenvolveu uma aplicação móvel que, a partir de dados obtidos dos sensores do smartphone, poderia definir a localização do usuário e a partir disso realizar ações no ambiente e no dispositivo, levando em conta a agenda do usuário. Já [Caetano et al. 2015] desenvolveu uma proposta de um módulo que seria uma extensão do UbiPri, o qual enviaria notificações relevantes aos usuários com base nas informações do ambiente que o usuário se localiza e na sua agenda.

Este trabalho, assim como o de [Nunes 2013], busca prover um melhor gerenciamento da privacidade do usuário, e com uma proposta parecida com a de [Caetano et al. 2015], apresenta o desenvolvimento de um módulo para o middleware UbiPri baseado no modelo taxonômico de [Leithardt et al. 2013], porém com o propósito e contribuição o gerenciamento do perfil dos usuários em ambientes ubíquos, não abordados nos trabalhos anteriores.

3. Módulo PRIPRO

Para o desenvolvimento do módulo PRIPRO foram utilizados serviços web com o modelo arquitetural REST, este recebe requisições de dispositivos que desejam autenticar usuários no ambiente, que podem ser: leitores biométricos, leitores de RFID (*Radio Frequency IDentification*), smartphones, entre outros. O serviço web informa ao dispositivo o perfil do usuário, dados do ambiente, recursos e serviços disponíveis utilizando o JSON (*JavaScript Object Notation*). O perfil do usuário é definido a partir dos dados que estão disponíveis para o módulo, esses dados são informações do usuário enviadas pela requisição e obtidos dos bancos de dados específicos da esfera em que ele se localiza.

A Figura 1 apresenta exemplos em que o serviço web deverá buscar dados de esferas diferentes, por exemplo, se um coordenador de curso da universidade for utilizar a área restrita aos funcionários da farmácia, ele deve ser barrado, visto que o status dele na universidade de nada afeta o seu perfil na farmácia. Uma esfera representa um conjunto de ambientes em uma determinada área, como por exemplo uma farmácia ou uma universidade.

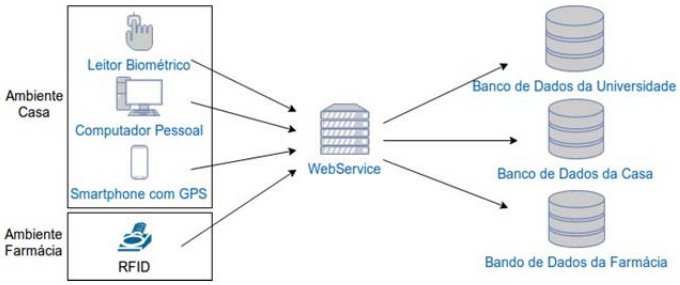


Figura 1. Arquitetura de Utilização do Serviço Web

Cada esfera possui as suas regras e particularidades, fazendo com que cada uma delas esteja em um contexto totalmente diferente. Para que uma requisição ao sistema seja válida há a necessidade de indicar certos parâmetros que permitam a busca dos dados do usuário. Visando a simplicidade da requisição, foram definidos como parâmetros o identificador do dispositivo do usuário, o que ele pretende fazer (entrar ou sair do ambiente) e o identificador do dispositivo autenticador. Deste modo, os dados necessários nos dispositivos são mínimos, simplificando a implementação.

As requisições no PRIPRO originam um procedimento de cinco etapas: 1) Busca dos dados do ambiente e da esfera em que o usuário se localiza; 2) Busca de informações no banco de dados específico da esfera; 3) Definição do perfil do usuário; 4) Inserção dos dados relevantes desta requisição no histórico; 5) Estruturação dos recursos e serviços para que sejam retornados no formato JSON, contendo os dados do ambiente, o perfil do usuário e os recursos e serviços disponíveis.

A Figura 2 apresenta as verificações que o PRIPRO faz na etapa 3, elas correspondem às regras para definição de cada perfil no cenário de testes de uma universidade, utilizado para testes e validação, escolhendo o perfil que mais se adapta ao usuário.

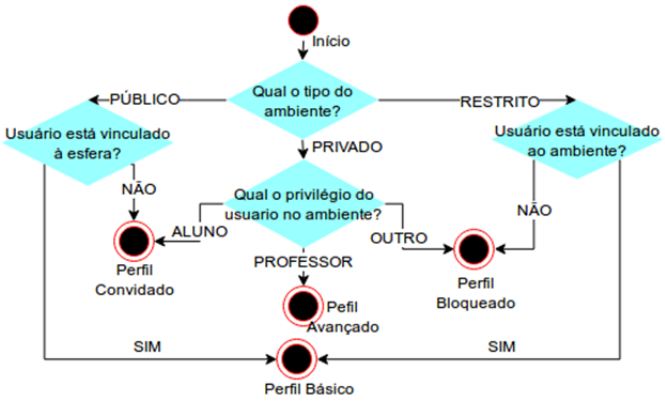


Figura 2. Definição de perfis num cenário de testes de uma universidade

Para que sejam definidos os perfis dos usuários nos ambientes individualmente, foram prototipadas várias esferas, cada uma delas com regras pré-definidas para a definição de cada perfil baseado no tipo do ambiente e no vínculo do usuário com a esfera. Todos os critérios e definições de parâmetros utilizados se fundamentaram em ambientes ubíquos com características de localização pública, privada e restrita. Em cada ambiente um tipo de perfil do usuário é atribuído de acordo com a sua localização e atribuição ao ambiente de acordo com o turno, dia, semana, mês relacionado.

4. Conclusão e Trabalhos Futuros

O módulo PRIPRO apresentado mostrou, nas implementações preliminares, eficiência na definição dos perfis, visto que ele pode ser acessado por qualquer dispositivo que disponha de acesso à internet e sempre terá um resultado conforme o esperado. Além disso, ele pode ser adaptado para qualquer esfera ou ambiente, centralizando todas as requisições que pretendem buscar o perfil do usuário.

Considerando que cada ambiente tem as suas regras e particularidades, foi constatado, nos testes parciais, que este sistema não consegue ser muito escalável, visto que

deve-se realizar a implementação de um novo módulo a cada esfera que tenha um banco de dados fora dos padrões definidos para cada ambiente individualmente. As regras, critérios e parâmetros foram fundamentadas no UbiPri, com isso várias atribuições e variáveis já estavam pré-definidas diminuindo assim a complexidade. Também foi possível utilizar a taxonomia de privacidade utilizada no middleware UbiPri, não necessitando elaborar ou definir funções que são primordiais em ambientes ubíquos, como por exemplo os itens necessários relacionados a comunicação.

Portanto, em trabalhos futuros, pretende-se realizar outros testes para viabilizar a escalabilidade do sistema. Para tanto, deve-se considerar a implementação de técnicas de inteligência artificial para auxiliar na definição de vários perfis simultâneos a partir dos dados fornecidos pelo banco de dados de cada ambiente. Com isso, será possível também trabalhar com parâmetros que permitam a evolução dos perfis dos usuários nos ambientes distribuídos em maior escala.

Além disso, será estudada a possibilidade da utilização de um novo perfil de ambiente personalizado, não abordado no trabalho de [Leithardt et al. 2013], que servirá para casos específicos em que um usuário comum poderia, temporariamente, obter um perfil mais avançado.

Este projeto foi desenvolvido com o apoio do Edital 01/2017 Programa de Bolsas Universitárias de Santa Catarina – UNIEDU BOLSAS DE PESQUISA - ARTIGO 170 - Universidade do Vale do Itajaí - UNIVALI.

Referências

Caetano, A. R. L., Bordin, M. V., Kolberg, W., B., B. G., Bianco, G. D., and Leithardt, V. R. Q. (2015). Uma proposta ubiservice para tratamentos de serviços direcionados a ubipri. In *13 Escola Regional de Redes de Computadores*. ERRC 2015, SBC.

Cherkaoui, A., Bossuet, L., Seitz, L., Selander, G., and Borgaonkar, R. (2014). New paradigms for access control in constrained environments. In *2014 9th International Symposium on Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC)*, pages 1–4.

Corrad, A., Montanari, R., and Tibaldi, D. (2004). Context-based access control management in ubiquitous environments. In *Third IEEE International Symposium on Network Computing and Applications, 2004. (NCA 2004). Proceedings.*, pages 253–260.

Leithardt, V., Borges, G., Rossetto, A., Rolim, C., Geyer, C., Correia, L., Nunes, D., and Sá Silva, J. (2013). A privacy taxonomy for the management of ubiquitous environments.

Nunes, B. R. (2013). An automation system for ubiquitous computing. Trabalho de Conclusão de Curso (Graduação em Ciências da Computação) - Universidade Federal do Rio Grande do Sul, Porto Alegre, 2013).

Satyanarayanan, M. (2017). The emergence of edge computing. *Computer*, 50(1):30–39.

Thakare, S., Patil, A., and Siddiqui, A. (2016). The internet of things – emerging technologies, challenges and applications. *International Journal of Computer Applications*, 149(10):21–25.

Analysis of Data Transmission Using RSA Encryption in Power Line Communication

Iago S. Ochôa, Douglas A. Santos, João A. Martins, Valderi R. Q. Leithardt

Laboratory Of Embedded and Distributed Systems – LEDS – Universidade do Vale do Itajaí (UNIVALI)
CEP 88302-202 – Itajaí– SC – Brazil

{iago.ochoa,douglasas,joao_martins}@edu.univali.br, valderi@univali.br

Abstract. *This paper describes a simulation model for transmission of encrypted data in power lines. The model was developed in MATLAB and consists of using asymmetric key cryptography to encode the data to be transmitted. The technique used to perform data transmission was the Power Line Communication technique.*

1. Introduction

Nowadays several organizations manufacture microcontroller units with specific applications of the smart grid segment. These applications must follow rules that govern this segment. The problem of these standards is that they designate the use of symmetric key cryptography in data transmission. With increasing processing power over the years it has been realized that a more secure type of encryption can be used in these systems.

With the emergence of smart grid networks, it has become possible to create more efficient, profitable and sustainable systems. With the advent of these new systems also appear new security breaches that can compromise them. Current literatures show that there is a large gap in security of data transmission in smart grid networks [Ali et al., 2015].

As a simulation scenario, a model of a point-to-point data transmission will be developed using the power line communication technique to send the data through the energy line. The asymmetric key cryptography chosen was RSA [Mathworks, 2012] because it is a widely used type in current days. The system will consist of transmitting an encrypted data from one point to another using the Gaussian Minimum Shift Key modulation. The integrity of received data will be verified after the transmission.

This paper will be structured in the following way: section 2 presents the related works, section 3 shows the proposed project model, section 4 presents the preliminary results obtained and section 5 the conclusions and future works.

2. Related Works

Table 1 shows a comparison of the related works. It is noticed that Lin, Latchman and Lee (2002) used the HomePlug protocol to model their system, the type of encryption described by this protocol is the Default Encryption Standard. Augusto (2011) used the PRIME protocol that describes the use of the Advanced Encryption Standard 128-bits. In the simulation model of Cataliotti, Di Cara, Fiorelli and Tiné (2012) and Chiotellis

and Cotis (2016), no protocol was used in the transmission model, and consequently no encryption.

Author	Characteristics		
	Protocol used	***Type of encryption used***	***Algorithm used.***
Lin, Latchman and Lee. (2002).	HomePlug	Symmetric	DES
Augusto. (2011)	PRIME	Symmetric	128-bit AES
Cataliotti, Di Cara, Fiorelli and Tinè. (2012)	None	None	None
Chiotellis and Cotis. (2016)	None	None	None
Ochôa and Leithardt. (2017)	None	Asymmetric.	RSA

Table 1. Protocol and cryptography comparative.

It is worth mentioning that our work proposes a transmission model without a defined protocol. The cryptography used will be the RSA asymmetric key, it was chosen because the microcontrollers aimed to the smart grid segment supports only symmetric key cryptography due their processing power, and our work intend to verify if it is possible to use asymmetric key cryptography in this microcontroller units. The modulation used to transmit data is the Gaussian Minimum Shift Key [MATLAB, 2016].

3. Suggested Model

The model was implemented in MATLAB because it has a ready-made block of functions for signal modulation in the frequency domain. Two codes were created separately, the first one consists in modulating and transmitting the signal, the second one is responsible for encrypting the data. After validating the two codes, both were integrated in one to make the system. Figure 1 shows the main steps of the proposed system and Figure 2 shows the main algorithm for the suggested model.

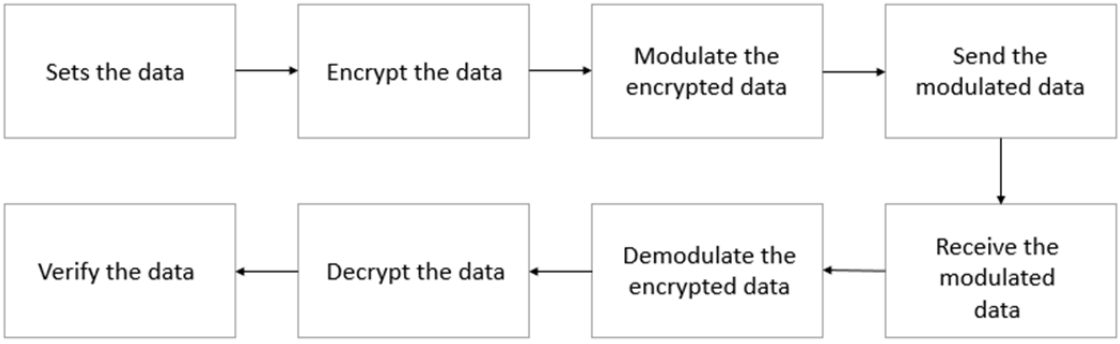


Figure 1. System modeling by author.

```
for i = 1 : 47                                  % 47 transmissions
    for j = 1 : 10                              % 10 bits each value
       aux(j)=(statemod(i,j));
    end
    a = vec2mat(aux,1);                         % vector to matrix
    data = double (a);                          % convert data to double
    modSignal = step(hMod, data);               % modulate signal
    receivedData = step(hDemod, modSignal);     % send data
     for j = 1 : 10
       staterec(i,j)=(receivedData(j));         % decrypt vector
     end
end
```

Figure 2. Main algorithm.

The code consists in transmitting a user-defined random data, after that this data is modulated in the frequency domain (GMSK modulation) – Gaussian Minimum Shift Key [Radio Electronics, 2008]. To simulate the transmission line, a white noise is added on it, to thereby generate a noise in the transmitted signal. Figure 3 shows the block diagram of the algorithm in SIMULINK [Mathworks, 2008].

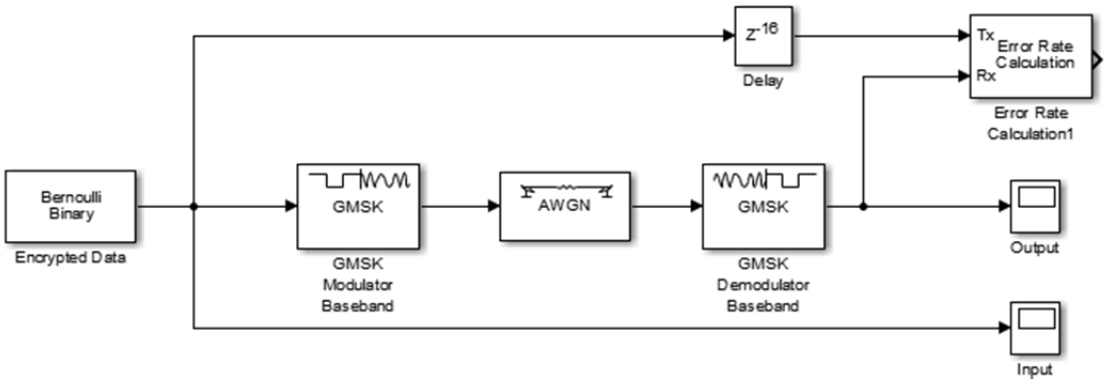


Figure 3. Block diagram of the main algorithm in SIMULINK by author.

4. Preliminary results

Executing the above code, it was obtained an unsatisfactory result for the suggested model. As shown in Figure 4, it is noticeable that the data received is totally different from the one sent. This is due to the integration of encryption and transmission codes. The main reason for this error in the decryption of the transmitted data is the delay generated by the modulation/demodulation process.

```
Public key: 3
Private key: 427
Message: 00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff
Cipher Text of the entered Message:ŏŏ  ęę  íí  ÝÝ  óó  ŎŎ  ıı  ǩǩ  jj  ǨǨ  ǏǏ  òò  KK  ЯЯ  □□  ŶŶ
Decrypted Message is:      □□ □□    □□ □□ □□ □□    □□ □□ □□    □□
```

Figure 4. Transmitted and received data.

5. Conclusions and Future Works

With the development of the work up to the present moment it was noticed that the result obtained was not satisfactory. The delay generated by the modulation/demodulation process compromised the transmission. It was due to this that the data received, and later decrypted, was not the same as the one transmitted.

It was noticed that to fix this problem it is necessary to use a communication protocol that meets the characteristics of the system, only with that will be possible to verify the integrity of the transmitted data.

The cryptography used has a short key due the limited processing power of microcontrollers for the smart grid segment. Extensive keys made the simulations impractical due to data encryption/decryption time.

In future is intended to apply the communication protocol aimed to this segment, to improve the transmission line model to become as close to real as possible and to apply performance characteristics of microcontrollers aimed to this segment to check system performance.

6. References

Ali, H., Amin, F., Hamid, J. and Alireza, G. (2015) "Security and Feasibility of Power Line Communication System". In: International Conference On Global Security, Safety and Sustainability", England.

MATHWORKS. (2012) "Implementation of RSA Algorithm", https://www. mathworks.com/matlabcentral/fileexchange/implementation-of-rsa-algorithm, July 2017.

MATLAB. (2016) "Modulate using GMSK method", https://www.mathworks.com /help/comm/ref/comm.gmskmodulator-class.html, May 2017.

Radio Electronics. (2008) "Gaussian Minimum Shift Key Method", http://www.radio-electronics.com/info/rf-technology-design/pm-phase-modulation/what-is-gmsk-gaussian-minimum-shift-keying-tutorial, April 2017.

MATHWORKS. (2008) "GMSK Mod/Demod", https://www.mathworks.com/ matlabcentral/newsreader/view_thread/169264, May 2017.

Augusto, R. (2011) "Simulation of Powerline Communication (PLC) for Smart Grid in OMNeT++", https://fenix.tecnico.ulisboa.pt/downloadFile/395145996565/resumo.pdf, June 2017.

Lin, Y., Latchman, H. and Lee, M. (2002) "A Power Line Communication Network Infrastructure for The Smart Home". In: IEEE Wireless Communications, p. 104-111, Switzerland.

Cataliotti, A., Di Cara, D., Fiorelli, R. and Tine, G. (2012) "Power-Line Communication in Medium-Voltage System: Simulation Model and Onfield Experimental Tests". In: IEEE Transactions on Power Delivery, p. 90-102, Switzerland.

Sendin, A., Penã, I. and Angueira, P. (2014) "Strategies for Power Line Communication Smart Metering Network Deployment", In: Energies, Switzerland.