Virtualização de redes de computadores – uma abordagem focada em mobilidade e georreferenciamento

Paulo Henrique Moreira Gurgel

Virtualização de redes de computadores – uma abordagem focada em mobilidade e georreferenciamento

Paulo Henrique Moreira Gurgel

Orientadora: Profa. Dra. Kalinka Regina Lucas Jaquie Castelo Branco

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências - Ciências de Computação e Matemática Computacional. *EXEMPLAR DE DEFESA*

USP – São Carlos
Setembro de 2014

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

G978v

Gurgel, Paulo Henrique Moreira
Virtualização de redes de computadores ? uma abordagem focada em mobilidade e georreferenciamento / Paulo Henrique Moreira Gurgel; orientadora Kalinka Regina Lucas Jaquie Castelo Branco. -- São Carlos, 2014.
101 p.

Dissertação (Mestrado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2014.

1. Netkit. 2. Virtualização de redes. 3. Redes de computadores. 4. User Mode Linux. 5. Ensino de redes de computadores. I. Branco, Kalinka Regina Lucas Jaquie Castelo, orient. II. Titulo.

Agradecimentos

Inicialmente, aos meus pais, Francisco César (*in memorian*) e Maria de Fátima, pelo enorme incentivo e esforço para que eu sempre me dedicasse a minha formação, desde a educação básica, de modo a viabilizar que eu pudesse ter uma formação acadêmica de nível superior, oportunidade que os mesmos não tiveram.

Agradeço também a minha esposa Rebeca, pela paciência, compreensão e por me acompanhar nas inúmeras atividades referentes ao desenvolvimento deste projeto e pela maravilhosa família que estamos formando.

Agradeço imensamente à amizade e orientação da Profª Drª Kalinka Regina Lucas Jaquie Castelo Branco, pelo companheirismo, orientação e pela parceria que se iniciou na graduação e se estendeu à pós, permitindo que eu tivesse hoje uma consciência acadêmica ampla, sólida e ética, viabilizando a execução deste projeto de mestrado.

Aos inúmeros colegas de pós-graduação dos diversos laboratórios de pesquisa do ICMC, LRM, LaSDPC que me acolheram, seja por atividades ou por pela companhia ao café, e em especial aos do LSEC, onde minhas atividades foram desenvolvidas. Aos colegas Daniel Pigatto, Natássya da Silva, Rayner Pires, Douglas Rodrigues, Júlio Rodrigues, Artur Avelar, companheiros de laboratório e atividades.

Aos companheiros de república, pelas horas de descontração e laser, Bruno Guazzelli, João Paulo Orlando e Felipe Bethonico, e aos diversos amigos de graduação que ainda mantenho contato.

Aos docentes do ICMC, pelos inúmeros seminários e pelas disciplinas cursadas. Aos coordenadores do programa Prof. Dr. Paulo Cesar Masiero e Prof.ª Dr.ª Maria da Graça Pimentel pelo excelente trabalho que viabilizam o nível de excelência do programa.

Agradeço imensamente a CAPES pela bolsa no início do curso, e a FAPESP pelo financiamento do projeto **2012/14867-3**, sem os quais este projeto não poderia ser realizado. E finalmente, ao Instituto de Ciências Matemáticas e de Computação e à Universidade de São Paulo pela excelente estrutura disponibilizada para que este projeto pudesse ser realizado.

Resumo

Ferramentas de virtualização de redes podem ser utilizadas com a finalidade de criar experimentos envolvendo redes de computadores e sistemas distribuídos em diversas aplicações, seja este para avaliação e validação de um sistema distribuído, teste de novas configurações em um ambiente de produção, ou mesmo para uso educacional. Várias são as opções de ferramentas disponíveis para a realização de tais experimentos. Entretanto, as ferramentas disponíveis não oferecem um suporte adequado a virtualização de redes móveis. O presente trabalho de mestrado apresenta uma proposta para a modificação de uma ferramenta de virtualização de código livre chamada *Netkit*, de modo a viabilizar a criação de experimentos que envolvam redes móveis. Neste caso, a ferramenta prevê que os dispositivos possam ser localizados em um espaço virtual e deslocados, de modo que variações como intensidade de sinal, trocas de pontos de acesso e experimentos que envolvam a localização espacial dos dispositivos possam ser efetuados.

Abstract

Network virtualization tools can be used to create several experiments like distributed systems evaluation and validation, test new software configuration in a virtual production environment or educational research. Although many tools enable a user to execute such experiments, few have mobile network support, presenting only wired network. The current project aims to present a proposal to modify a open source virtual network emulator called *Netkit*, enabling mobile network experimentation. The modified tool is capable to create virtual devices that can be located in a virtual space and moved along it, providing signal intensity variation, handovers, and able to address experimentation scenarios that requires spatial and geodesic location of the virtual mobile devices.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Siglas

ACK	*Acknowledge*
ACL	*Access Control List*
ARP	*Address Resolution Protocol*
ASCII	*American Standard Code for Information Interchange*
ASN	*Autonomous System Number*
BGP	*Border Gateway Protocol*
CEP	Código de Endereçamento Postal
CISC	Centro de Informática de São Carlos
CMS	*Conversional Monitor System*
CSMA	*Carrier Sense Media Access*
CSMA/CA	*Carrier Sense Media Access with Collision Avoidance*
CSMA/CD	*Carrier Sense Media Access with Collision Detection*
CRC	*Cyclic Redundancy Check*
DHCP	*Dynamic Host Configuration Protocol*
DIT	Departamento de Ingeniería de Sistemas Telemáticos
DNS	*Domain Name Server*
DUAL	*Diffusing Update Algorithm*
EIGRP	*Enhanced Interior Gateway Routing Protocol*
FQDN	*Fully Qualified Domain Name*
FTP	*File Transfer Protocol*
Gb	*Gigabytes*
GPL	*General Public Licence*
GPS	*Global Positioning System*
GRE	*Generic Routing Encapsulation*
HTTP	*Hypertext Transfer Protocol*
ICMC	Instituto de Ciências Matemáticas e Computação
ICMP	*Internet Control Message Protocol*
IIS	*Internet Information Service*
ISC	*Internet Systems Consortium*
IMAP	*Internet Message Access Protocol*
IP	*Internet Protocol*
IPv4	*Internet Protocol* versão 4
IPv6	*Internet Protocol* versão 6
ISO	*International Organization for Standardization*
ISP	*Internet Service Provider*
MAC	*Media Access Control*
Mbytes	*Megabytes* (Unidade de armazenamento equivalente a 1016 bytes)
MIB	*Management Information Base*

MIME	*Multipurpose Internet Mail Extension*
MPLS	*Multiprotocol Label Switching*
MTA	*Mail Transport Agent*
NAT	*Network Address Translation*
NetML	*Network Markup Language*
NIC	*Network Interface Card*
OID	*Object Identifier*
OSI	*Open Systems Interconnection*
OSPF	*Open Shortest Path First*
PAE	*Programa de Atividade de Estágio*
PDF	*Portable Document Format*
PHP	*PHP Hypertext Preprocessor*
POP	*Post Office Protocol*
RAM	*Random Access Memory*
RFC	*Request for comments*
RIP	*Routing Information Protocol*
ROM	*Read Only Memory*
SMTP	*Simple Mail Transport Protocol*
SNMP	*Single Network Management Protocol*
SPDU	*Session Protocol Data Unit*
SSL	*Secure Socket Layer*
SYN	*Syncronize*
TCP	*Transport Control Protocol*
TPDU	*Transport Protocol Data Unit*
TTL	*Time to live*
UDP	*User Datagram Protocol*
ULBRA	Universidade Luterana do Brasil
UML	*User Mode Linux*
UPM	*Universidad Politécnica de Madrid*
USP	Universidade de São Paulo
VNUML	*Virtual Network User Mode Linux*
x64	Arquitetura de processadores de 64 bits similares ao x86
x86	Arquitetura de processadores baseados no Intel 8086
XML	*Extensible Markup Language*

Capítulo

1

1. Introdução

1.1. Considerações iniciais

Neste capítulo, serão apresentadas contextualização, motivação e objetivos do projeto. A contextualização descreve o cenário de virtualização de redes e seu uso no ensino de redes de computadores. São descritas a motivação para o desenvolvimento e explicadas as decisões iniciais tomadas. Finalmente, são apresentados os objetivos do presente projeto de mestrado. Também neste capítulo é esclarecido como a solução proposta se diferencia das existentes e quais as contribuições técnicas e acadêmicas obtidas.

1.2. Contextualização e Motivação

O conceito chave deste trabalho é a experimentação de cenários de configuração de redes com algum grau de complexidade. Cenários estes, que permitam testar diferentes implementações de protocolos, ferramentas e configurações, bem como verificar o comportamento dos componentes da rede em situação de carga. Neste tipo de experimentação têm-se dois tipos de problemas: inicialmente tem-se a aplicação ao processo de ensino e aprendizagem de redes de computadores, onde limitações no ambiente disponível podem restringir a possibilidade de aplicações de atividades práticas; e tem-se também a concorrência pelo uso dos equipamentos e a inviabilidade de se ter laboratórios completos com a presença de todos os equipamentos necessários para a realização de tais experimentos.

Assim, em uma turma com diversos alunos, a concorrência pelo uso dos poucos equipamentos pode inviabilizar a experiência prática. Se o ambiente, no entanto, for compartilhado com outras tarefas, fornecer permissões de configuração para os aprendizes pode trazer problemas para a realização das atividades posteriores (RIMONDINI, 2007). De modo análogo, considerando redes empresariais, realizar experimentos dentro em um ambiente em produção pode causar transtornos aos negócios devido à redução da qualidade ou interrupção dos serviços prestados (BARBOSA, ANJOS e BOGO, 2009).

Para permitir que a experimentação quando não existe a viabilidade para aquisição de equipamentos que possibilitem a construção dos cenários de rede que precisam ser estudados, grupos de pesquisa e empresas desenvolveram ferramentas de software que permitem a aplicação de tais experimentos. É possível dividir estas ferramentas em duas categorias:

a. Simulação: Utilizam softwares que permitem a configuração do ambiente por meio de objetos que representem os elementos da rede. Nesta categoria, o funcionamento da rede é reproduzido por meio de regras programadas baseadas no ajuste sobre os objetos presentes na rede. Os simuladores tipicamente fornecem um conjunto limitado de funcionalidades, podendo ser criados, inclusive, para simular apenas um tipo de cenário. Este é o caso do C-BGP (QUOITIN, 2012), que permite a criação de uma rede de roteadores e sistemas autônomos com parâmetros para o cálculo das tabelas de roteamento utilizando o protocolo BGP (*Border Gateway Protocol*). Um simulador mais genérico é o OPNET Modeler (OPNET, 2012), que possui interface gráfica de experimentação e o suporte a uma quantidade maior de protocolos e métricas de funcionamento. Outro exemplo de simulador é o Bosom NetSim (BOSOM, 2012), que permite a configuração de redes com diversos modelos de roteadores comerciais. É importante destacar que, por mais avançados que sejam estes simuladores, eles ainda funcionam dentro das restrições às quais foram desenvolvidos. Isso limita, em algum nível, a possibilidade de experimentação, seja na utilização de softwares e protocolos não difundidos, seja por não permitir o uso das mesmas ferramentas que serão utilizadas no ambiente de produção.

b. Virtualização: Para remover as restrições funcionais impostas pela simulação, é possível realizar a virtualização da rede. Na virtualização da rede, máquinas virtuais são instanciadas e enlaces, também virtuais, são criados. A escalabilidade da rede é restrita pela disponibilidade de memória RAM do sistema hospedeiro, uma vez que parte da memória total deverá ser reservada para cada instância virtual em execução. É possível realizar virtualização completa por meio de ferramentas como o *Oracle Virtual Box* (ORACLE, 2014) ou *VMWare Player* (VMWARE, 2014), sem no entanto ter controle sobre a topologia de rede criada. Outra solução utiliza softwares como o *Netkit* (BATTISTA et. at., 2012), VNUml (Virtual Network User Mode Linux) (FERNÁNDEZ e GÁLAN., 2012) e *Gini (GINI is not Internet)* (MAHESWARAN et. al.., 2009), todos estes baseados no software *User Mode Linux* (UML) (DIKE, 2006). O UML permite que uma máquina virtual completa seja instanciada como um processo, utilizando recursos do sistema operacional hospedeiro para comunicação com o *hardware*. Além disso, os cenários que podem ser montados por meio do UML são maiores, pois os requisitos da virtualização são bem menores (RIMONDINI, 2007).

Embora as ferramentas baseadas em virtualização permitam o desenvolvimento de vários cenários de redes cabeadas, estas ferramentas falham quando é necessário realizar a criação de cenários de rede sem fio e mobilidade. De fato, a virtualização UML elimina do experimento métricas e características físicas da rede, onde o enlace físico deve, então, ser considerado como transparente. Dentre as ferramentas baseadas em UML, dois projetos destacam-se na tentativa de prover experimentações em ambientes sem fio. O projeto *GINI* (*GINI is not Internet*, GINI não é a Internet), cuja página do projeto na Internet não recebe atualizações desde 2009 (GINI, 2009) possui um módulo para que a conexão entre os nós seja feita por meio de interfaces de redes sem fio, no padrão 802.11, e o projeto ORBIT, que possui uma infraestrutura fixa de 400 nós e interfaces de rede 3G e 802.11 fixas que podem ser utilizadas para experimentação (ORBIT, 2013).

Tais ferramentas, entretanto, não permitem reproduzir experimentos onde os dispositivos estejam em movimento, deslocamento este que gera necessidade de trabalhar com a variação da intensidade do sinal e a troca de ponto de acesso, uma operação conhecida como *handover*. A ausência do *handover* limita a quantidade de experimentos que podem ser realizados, quando considerada toda a estrutura tecnológica das redes móveis e, fornece a motivação necessária para a realização deste projeto.

1.3. Objetivos

Este projeto teve por objetivo a extensão da ferramenta *Netkit*, que permite a virtualização de redes, de modo a incluir funcionalidades que viabilizem a realização de experimentos que envolvam os protocolos mais recentes de comunicação sem fio, inclusive com endereçamento IPv6, rotas com múltiplos pontos de acesso, e troca de ponto de acesso, operação conhecida por *handover*, considerando a mobilidade dos nós de tais redes. Tais funcionalidades estão sendo desenvolvidas com foco no uso do *Netkit* como *objeto de aprendizagem* no contexto do ensino de redes móveis de computadores. Para obter uma aproximação da realidade, os nós são georreferenciados, isto é, suas coordenadas no espaço são conhecidas, bem como seu movimento é descrito e o impacto no sinal, reproduzido.

Adicionalmente, um conjunto de experimentos educacionais foi desenvolvido de acordo com as funcionalidades implementadas. Tais experimentos são apresentados na forma de tutoriais que envolvem o uso de ferramentas de virtualização de redes, tais como os descritos em GURGEL et. al. (2012). Outros tutoriais serão desenvolvidos em trabalhos futuros, por

alunos de iniciação cientifica e trabalhos de conclusão de curso, aos moldes de anteriores já desenvolvidos com o *Netkit* (RODRIGUES, 2013) e (GURGEL, 2010).

1.4. A escolha do Netkit

Para a escolha da ferramenta estendida, alguns critérios precisaram ser observados. Em resumo:

- ✓ Ferramentas de virtualização de redes
- ✓ Configuração livre da topologia
- ✓ *Open Source* ou com API (*Application Programming Interface*) de extensão
- ✓ Compartilhamento dos experimentos
- ✓ Baixo consumo de recursos

Não foi considerado para este trabalho, pelos vários motivos explicados na contextualização, o uso de simuladores, pois embora possam fornecer pontos de vista interessantes, o objetivo é que a experimentação seja feita utilizando softwares reais, disponíveis no mercado. São exemplos destes softwares, *Apache Web Server* da *Apache Software Foundation (APACHE, 2014)*, *BIND (Berkeley Internet Name Domain)* da *Internet Systems Consortium (ISC, 2014)* e similares. Em momentos futuros do presente texto, estes softwares serão denominados apenas como Apache[1] e Bind[2], assim como são conhecidos popularmente. O aluno também pode desenvolver seus softwares aplicativos e servidores, na linguagem de sua preferência[3] e validar no ambiente proposto.

A ferramenta precisaria permitir a livre configuração da topologia, requisito onde softwares de virtualização completa, como *Xen* (XEN PROJECT, 2014) e *VirtualBox* (ORACLE, 2014) falham. Nas ferramentas de virtualização tradicional, todas as interfaces de rede são diretamente expostas ao hospedeiro da máquina virtual, o que não é sempre o caso da virtualização de redes consideradas.

[1] Apache Web Server sendo referenciado apenas como Apache - http://w3techs.com/technologies/details/ws-apache/all/all - acessado em 10/09/2014

[2] Bind apresentado pelo desenvolvedor como o sistema de servidor DNS mais utilizado - http://www.isc.org/downloads/bind/ - acessado em 10/09/2014.

[3] O sistema de arquivos padrão do NetkitG3 acompanha as ferramentas de desenvolvimento para Python, C/C++, PHP e Perl. O sistema de arquivos denominado jumbo, contém também Java, Ruby e FreePascal.

Sendo a extensão de uma ferramenta, somente poder-se-ia considerar um software cujo código fonte fosse aberto, ou que fornecesse uma interface de programação evidente para a extensão. Não estender significaria desenvolver funcionalidades de rede com fio já presentes em outras ferramentas consolidadas. Durante este trabalho, nos referiremos a esta propriedade como código livre, de acordo com a filosofia FOSS[4] (Free and Open Source Software), onde um software possui o código fonte aberto à comunidade, sendo seu código livre para que o usuário possa estudá-lo, copiá-lo, modificá-lo de qualquer forma.

Outra característica importante é a possibilidade e o meio de viabilizar o compartilhamento de um cenário de experimentação. Neste caso, o professor deveria gerar um arquivo de configuração inicial, ou equiparado, e este arquivo poderia ser distribuído a seus alunos de modo que uma instância do mesmo cenário fosse carregada no computador do estudante, viabilizando a execução dos experimentos.

Também é desejado que a ferramenta consuma poucos recursos computacionais, em particular de memória e espaço em disco. É desejado que este consumo seja tão reduzido quanto possível, dado o direcionamento educacional do projeto e a possibilidade de execução de experimentos que envolvam redes de maior tamanho e complexidade. Neste aspecto, foi fundamental o trabalho de Fuertes e Vergara (2007) que demonstra uma comparação de várias ferramentas de virtualização que permitem, de alguma forma, experimentos em redes. Um resumo deste comparativo é exibido na Tabela 1.

Tabela 1: Comparativo de recursos das ferramentas de virtualização

Ferramenta	CPU@Startup (%)	Memória (MB)
VNUML	31,73	270,01
Netkit	48,16	185,82
Qemu	87,51	241,52
Xen	35,14	204,90
Vmware	85,49	286,76
Virtual Box	80,41	668,86

Por meio deste artigo, foi possível observar que a distribuição de cenários é uma característica atingida com sucesso pelas ferramentas *VNUML* e *Netkit*. O *Netkit*, no entanto, embora apresente maior utilização de processamento durante a inicialização de um experimento, consumiu apenas 68,8% da memória comparado ao VNUML.

Embora outros sistemas de virtualização de redes, como o VNUML, atingissem também todos os requisitos, o *Netkit* se sobressai em duas das características desejadas, na distribuição

[4] Desambiguando free e open, http://freeopensourcesoftware.org, acessado em 10/09/2014.

do cenário e no menor consumo de recursos, tornando-se o melhor candidato considerando estes critérios.

O *Netkit* é um emulador de rede de computadores, desenvolvido por pesquisadores da Universidade *Romatrè*, Itália, com código fonte aberto (RIMONDINI, 2007). Sua operação é simples, e permite o uso de máquinas virtuais *Linux*, interligadas por meio de enlaces virtuais onde os diversos experimentos e redes, denominados laboratórios, podem ser realizados de forma segura e sem custo (RIMONDINI, 2007). O *Netkit* é uma ferramenta utilizada para os objetivos almejados neste projeto, ou seja, seu uso educacional, porém sua funcionalidade é restrita as redes cabeadas. A distribuição de cenários pode ser feita por meio de um conjunto de arquivos de configuração, distribuídos tipicamente na forma de um pacote "tar". Não obstante, arquivos de configuração dos serviços das máquinas virtuais podem ser distribuídos e o *Netkit* substitui os arquivos originais pelas modificações distribuídas na primeira inicialização de cada máquina virtual.

A ferramenta *Netkit* já tem sido utilizada nas disciplinas de graduação do ICMC/USP, quando ministradas pela professora orientadora deste projeto, a partir do material de experimentação desenvolvido no trabalho de conclusão de curso e atividades posteriores do autor do presente trabalho. Além disso, o material desenvolvido tem sido, também, utilizado por docentes de diversas instituições de ensino, como o IFSP (Instituto Federal de São Paulo), UFRJ (Universidade Federal do Rio de Janeiro), UFJF (Universidade Federal de Juiz de Fora), UFMA[5](Universidade Federal do Maranhão).

1.5. Resultados obtidos

O artefato resultante é denominado ***NetkitG3***[6], em alusão aos 3 tipos de redes que a extensão passa a cobrir, cabeada, 802.11 e móvel 3G/4G. A arquitetura de desenvolvimento do software é modularizada, permitindo que novas extensões, inclusão de protocolos e modos de operação sejam feitos de forma facilitada em trabalhos futuros.

No escopo deste trabalho estão inclusos os módulos de *backend*, que controlam a mobilidade, posicionamento dos satélites e as mudanças de intensidade de sinal. Do *frontend*, módulos capazes de permitir visualizar a posição espacial das máquinas virtuais e a intensidade dos sinais sendo captados. Tais módulos estão descritos com detalhes no Capitulo 5. A

[5] Informação obtida por contato direto com os docentes e alunos destas instituições.
[6] G de Georreferenciado, e também de Gurgel, e 3 por cobrir os três tipos de comunicação mais populares. Não se trata de uma analogia direta ao 3G. Também será a terceira versão (major release) do Netkit.

intensidade do sinal pode ser modificada pelo ambiente de simulação de acordo com as características reproduzidas (intensidade, potência e ganho do transmissor, sensibilidade, distância e atenuação do meio).

A construção de um módulo adicional que permita a criação de *hosts* com sistemas embarcados viabiliza ampliar as possibilidades de experimentação. O exemplo proposto é um módulo sensor de temperatura, que possui uma interface de rede básica sem fio.

A validação foi feita por meio da criação de casos de uso, explorando diferentes aspectos e protocolos cobertos pela implementação da ferramenta, como segurança, mobilidade e fluxo de sinal. Desta forma, foi necessário reproduzir alguns cenários com equipamentos reais, notebooks, pontos de acesso sem fio e *modens* 3G, e a replicação do mesmo no *NetkitG3*.

Pelo uso didático, interfaces de criação, monitoração e controle dos experimentos receberam atenção especial. Sendo assim, a versão final pode ser utilizada de modo prático, permitindo que o usuário final possa investir seu tempo de estudo e pesquisa no experimento em si, na coleta de resultados e na tabulação, e não na configuração do ambiente. Estes módulos devem operar como objetos de aprendizado, viabilizando o uso em sala, com orientação do professor e material de apoio, sejam cursos presenciais ou à distância.

1.6. Estrutura do texto

O restante deste projeto está organizado da seguinte maneira:

O capítulo 2 apresenta uma revisão bibliográfica envolvendo os conceitos teóricos sobre redes móveis de computadores, incluindo uma introdução às redes de computadores em geral, enfatizando as diferenças tecnológicas entre as mesmas.

No capítulo 3, uma revisão teórica sobre os conceitos e teorias de georreferenciamento é descrita, relacionando a mobilidade com os experimentos que podem ser realizados em aplicações com informações sobre o espaço próximo.

O capítulo 4 descreve uma apresentação sobre os conceitos envolvendo as ferramentas de virtualização e simulação. Neste, serão descritas várias ferramentas de virtualização, as opções baseadas em UML e as características do *Netkit* que culminaram na sua escolha.

No capítulo 5, são apresentados os métodos e técnicas utilizados para o desenvolvimento deste trabalho, bem como a ferramenta *NetkitG3* e uma discussão sobre os resultados obtidos.

Finalmente, o capítulo 6 apresenta a conclusão do presente trabalho, onde são elencadas as contribuições e os trabalhos futuros.

Capítulo

2

2. Redes móveis

2.1. Considerações Iniciais

No decorrer deste capítulo são apresentados os conceitos teóricos e tecnológicos relacionados a redes móveis de computadores. Na seção 2.2, são apresentados conceitos gerais sobre o assunto redes de computadores, de modo a apresentar a terminologia, definições, principais elementos da arquitetura de redes abordadas neste trabalho. Na seção 2.3 uma revisão teórica sobre redes móveis é apresentada. Nessa são enunciadas as principais diferenças entre redes móveis e redes cabeadas tradicionais. Na seção 2.4, é apresentada uma revisão de trabalhos recentes na área de redes móveis, enumerando trabalhos que se aproximam do presente projeto de pesquisa. Finalmente a seção 2.5 apresenta considerações finais relacionando o conteúdo apresentado com o presente projeto.

2.2. Redes de computadores

2.2.1. Conceitos básicos

O Modelo *Open Systems* Interconnection (interconexão de sistemas aberto), conhecido geralmente como modelo OSI, definido pela *International Organization for Standardization* (ISO, Organização Internacional para Padronização) é uma arquitetura de sete camadas que define de que modo sistemas abertos podem ser interconectados pela rede de comunicação. O modelo começa pela camada física que trata da transmissão bruta dos pulsos elétricos (*bits*) e interpretação dos níveis eletrônicos. A segunda camada é a camada de enlace, do inglês *data link layer*, que tem como função o tratamento de erros e o envio de quadros de confirmação, além do controle de fluxo quando o transmissor é mais rápido que o receptor.

Para realizar essa coordenação, a sub-camada de acesso ao meio, conhecida como camada MAC (*Media Access Control*), faz uso de um protocolo de controle denominado CSMA/CD (*Carrier Sende Multiple Access with Collision Detection Protocol* – Protocolo de múltiplo acesso sensível a portadora com detecção de colisão). Neste protocolo, quando um dispositivo

deseja realizar uma transmissão ele o faz. Se dois nós da rede realizarem essa transmissão simultaneamente, haverá uma colisão no meio físico, neste caso nos cabos de rede, e por meio da leitura da intensidade do sinal as interfaces de rede conseguem detectar que houve colisão. Cada transmissor, então, deverá esperar um tempo aleatório para realizar uma nova tentativa. Este protocolo melhorou o desempenho de protocolos utilizados anteriormente em outros padrões de rede, como o ALOHA[7] e o *slotted* ALOHA, pois esses protocolos apresentavam ociosidade do canal ou grande taxa de colisões, enquanto o CSMA/CD consegue melhor aproveitamento do meio (TANEMBAUM, 2003).

Em seguida, a camada de redes controla a operação de sub-rede tratando o endereçamento e rotas para que os pacotes sejam corretamente transmitidos da origem ao destino. A quarta, a camada de transporte, cuida da transferência das informações circuladas pela rede para os processos em execução criando canais de comunicação fim a fim com ou sem garantia de entrega. Além disso, ela cuida do sequenciamento dos pacotes quando necessário. A camada de sessão é responsável pela manutenção de sessões para comunicação com usuários diferentes, controlando o diálogo e realizando a sincronização. A sexta, camada de apresentação, é responsável pela tradução entre a camada superior de aplicação e as camadas inferiores que se preocupam com a transmissão da informação, incluindo criptografia e padronização da codificação de caracteres. Finalmente, a camada de aplicação é a camada que apresenta os protocolos dos serviços que podem ser executados nos computadores (TANENBAUM, 2003).

Os computadores e outros elementos que participam de uma rede podem ser interligados por vários tipos de dispositivos. O primeiro dos dispositivos é o *hub* que permite a ligação de vários dispositivos de acordo com seu número de portas. O *hub* forma um domínio de colisão e apenas emite o sinal recebido em uma de suas portas para os demais computadores. Este fato dá a ele duas características: a primeira é que ele faz conexão por difusão, enviando a todos os computadores a informação recebida; a outra é a existência de colisões caso dois computadores tentem enviar pacotes simultaneamente.

Os *hubs* ativos podem amplificar o sinal de rede, trabalhando como repetidores. Para a camada de enlace têm-se as pontes (*bridges*) e os comutadores (*switches*). A *bridge* interliga duas redes diferentes analisando os endereços físicos no quadro. O *switch* é utilizado de forma análoga ao *hub*, ligando dispositivos individuais, segundo Tanenbaum (2003) foi comum utilizar

[7] Neste caso, ALOHA não é um acrônimo, mas o próprio nome do protocolo de controle. Foi preservada a grafia presente nas bibliografias que citam o protocolo. O nome do protocolo significa, literalmente, "Olá", saudação típica do Havaí, local da universidade (Universidade do Havaí) onde o mesmo foi desenvolvido.

os termos de forma intercambiável, mas há diferenças consideráveis entre um e outro. A principal diferença é que na *bridge* as redes tem domínios de colisão particulares e se não for necessário fazer o encaminhamento de uma rede para outra o quadro é descartado. Já no *switch* há o caminho único dos quadros e o encaminhamento é obrigatório (TANENBAUM, 2003).

Hoje no mercado os *hubs* são raros, já que existem os "*hub-switches*", que embora não tenham capacidade de configuração, são capazes de analisar os quadros na camada de enlace e encaminhar os mesmos para os destinatários corretos (MORIMOTO, 2008).

Para que essa comunicação ocorra de forma efetiva, é necessária a utilização de protocolos.

Protocolos são regras que definem a forma de comunicação entre dois elementos em rede. Vários protocolos definem a limitação do tamanho, estipulam códigos de controle de erros, estruturam os dados que precisam ser enviados e o quão rápido os dados podem ser enviados (ANDERSON e BENEDETTI, 2009). Enquanto o modelo OSI define uma boa arquitetura, ele não fornece protocolos práticos para a criação dos elementos de rede.

Os protocolos utilizados na Internet em geral fazem parte do modelo TCP/IP, que por sua vez não é tão utilizado como modelo de referência (TANENBAUM, 2003).

A camada de enlace trabalha realizando a comunicação por meio de quadros. Quadros são sequências de *bits* organizados para transportar a informação. A estrutura de um quadro é ilustrada na **Figura 1**.

Figura 1: *Quadro da camada de enlace*

Um quadro é composto das seguintes seções:

1. **Preâmbulo**: Uma sequência de sete bytes "10101010" para sincronia do *clock*.
2. **Inicio do quadro**: Um byte 10101011 indicando que a seguir vem conteúdo útil.
3. **Endereço MAC [8] destino**: Seis bytes com o endereço de destino do quadro.
4. **Endereço MAC origem**: Seis bytes com o endereço do remetente do quadro.

[8] Media Access Control - Controle de Acesso ao Meio

5. **Tipo de carga**: Dois bytes especificando o tipo de carga.
6. ***Payload***: O conteúdo efetivo sendo transmitido de 46 à 1500 bytes.
7. **CRC (*cyclic redundancy check*)**[9]: Um código de verificação de erros para verificar se o conteúdo do quadro está consistente.

Os equipamentos de transmissão que trabalham na camada de enlace utilizam o endereço MAC para endereçar o quadro a seu destino. O endereço MAC, composto de 48 *bits* e também conhecido por endereço físico ou endereço de LAN, é distribuído pelo Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE – *Institute of Electrical and Electronics Enginners*), que gerencia o espaço de endereços disponíveis aos fabricantes.

Cada placa tem um endereço MAC exclusivo gravado em sua ROM. Entretanto, na camada de redes a comunicação é feita por meio de endereços IP's (*Internet Protocol*) que são hierarquizados dependendo do local do mundo onde o computador conectado se encontra.

Desse modo, faz-se necessária uma tradução entre estes dois endereços (MAC e IP) e isso é feito pelo *Address Resolution Protocol* (ARP), definido pela RFC8262. Quando o computador destino da mensagem está no mesmo domínio de colisão o protocolo ARP envia um *broadcast*, uma mensagem destinada a todos os computadores do domínio, solicitando que o portador do IP desejado responda com seu endereço MAC. O computador que conhece este endereço responde então destinando ao computador de origem um pacote similar, com seu endereço de rede. O protocolo ARP então adiciona uma entrada referente àquele IP na tabela ARP da máquina origem e dessa forma pode enviar a mensagem original (KUROSE e ROSS, 2006). Na **Tabela 2** é ilustrado um exemplo da tabela ARP.

Tabela 2: Exemplo de tabela ARP

Endereço IP	Endereço MAC	Time to Live – TTL[10]
10.1.1.1	0D:0A:3F:45:2A:9B	00:07:35
10.1.1.2	0D:0A:3F:45:3B:81	00:16:02
10.1.1.4	3C:A4:2B:73:13:1C	00:18:47
10.1.1.5	FF:FF:FF:FF:FF:FF	00:00:00

Já na camada de redes, a comunicação é realizada por meio do protocolo IP (*Internet Protocol*, protocolo da Internet). O protocolo IP na sua quarta versão (IPv4) utiliza um endereço

[9] Cyclic redundancy check
[10] Tempo de vida, no caso do ARP, que a entrada permanece na tabela. Depois de passado o tempo a entrada será atualizada para verificar se o IP não pertence a outro endereço MAC

formado por 32 bits, geralmente dispostos na forma de 4 números decimais, conhecido como endereço IP.

Diferente do *MAC Address,* que é gravado na *Network Interface Card* (NIC – placa de interface de rede) pelo fabricante, o IP público, aquele que é visível na Internet, é atribuído hierarquicamente e depende do país e do provedor de serviço da Internet (ISP – *Internet Service Provider*).

A hierarquização do endereço IP é importante para o estabelecimento das rotas. Rotas são os caminhos percorridos pela informação entre vários nós da rede até ser entregue a seu destinatário, pois os endereços IPs são organizados de forma que se possa descobrir a localização de um computador pelo IP.

Parte do endereço IP é como o CEP (Código de Endereçamento Postal) que fornece uma localização generalizada, enquanto outra parte dele é como rua e número, que fornecem a localização precisa (CASAD, 2004).

Ao ser observado o conteúdo do *payload* de um quadro de enlace, tipicamente um pacote IP, é composto do *header* e seus dados. O cabeçalho de um protocolo IP é montado de acordo com a Figura 2.

Figura 2: *Estrutura do cabeçalho IP*

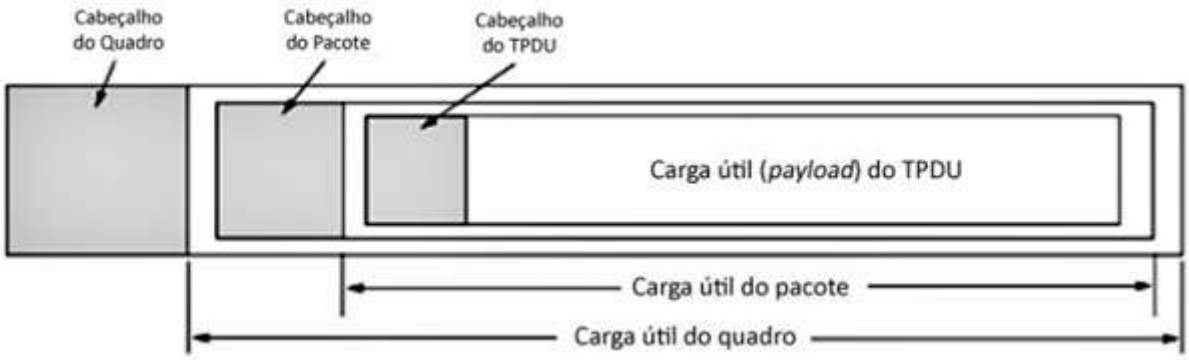
<table>
<tr><th rowspan="2">Deslocamento de bits</th><th colspan="3">0 a 15</th><th colspan="2">16 a 31</th></tr>
<tr><th>0 a 3</th><th>4 a 7</th><th>8 a 15</th><th>16 a 18</th><th>19 a 31</th></tr>
<tr><td>0</td><td>Versão</td><td>Tamanho do cabeçalho</td><td>Tipo de Serviço (agora DiffServ)</td><td colspan="2">Comprimento (pacote)</td></tr>
<tr><td>32</td><td colspan="3">Identificador</td><td>Flags</td><td>Offset</td></tr>
<tr><td>64</td><td colspan="2">Tempo de Vida (TTL)</td><td>Protocolo</td><td colspan="2">Checksum</td></tr>
<tr><td>96</td><td colspan="5">Endereço origem</td></tr>
<tr><td>128</td><td colspan="5">Endereço destino</td></tr>
<tr><td>160</td><td colspan="5">Opções (opcional)</td></tr>
<tr><td>192</td><td colspan="5">Dados</td></tr>
</table>

Os campos do cabeçalho IP são:

1. **Versão:** A versão do protocolo, com o valor 4.
2. **Tamanho do cabeçalho**: Multiplicador do tamanho do cabeçalho, geralmente 5 ou 6 se houver o campo opções. O valor será multiplicado por 32 para encontrar o tamanho em bits.
3. **Tipo de serviço**: Opções para interferir no tipo de comunicação, como pedido de alta prioridade e aviso de congestionamento (ANDERSON e BENEDETTI, 2009). De acordo com a RFC 2474[11], no entanto, foi substituída pelo campo serviço diferencial que classifica o pacote de modo a instruir os roteadores a tomarem medidas de qualidade.
4. **Comprimento do pacote**: Informa o tamanho total do conteúdo, independente da fragmentação.
5. **Identificador**: Número sequencial do pacote. Se for necessário ocorrer a fragmentação, todos as partes do mesmo pacote terão a mesma identificação para posterior reconstrução do pacote original.
6. ***Flags***: Solicitam do meio se o pacote pode, e em podendo como será fragmentado.
7. ***Offset***: Indica a sequência dos fragmentos de mesma identificação.
8. **Tempo de vida**: Representado em segundos ou número de saltos, indica quanto tempo o pacote pode sobreviver sem ser descartado. Isso impede que o pacote seja encaminhado indefinidamente entre roteadores sem encontrar seu destino final.
9. **Protocolo**: Um identificador que informa o protocolo da camada de transporte que a carga útil contém.
10. ***Checksum*** (Soma de verificação): É o número de 16 bits calculado que contém o código de verificação da integridade do cabeçalho.
11. **Endereço origem**: É o endereço IP de origem do pacote.
12. **Endereço destino**: É o endereço IP do destinatário que deve receber o pacote
13. **Opções**: Campo opcional que pode conter informações de segurança, carimbo de data e hora, informações sobre roteamento específico, informações para depuração ou teste e outros usos. O campo *Options* pode ter um subitem denominado ***Padding*** que indica a existência de mais campos opções (CASAD, 2004).
14. **Dados**: A carga de dados útil, geralmente um *Transport Protocol Data Unit* (TPDU – unidade de dados de protocolo de transporte).

[11] http://tools.ietf.org/html/rfc2474 - acessado em 10/08/2014

Figura 3: *Aninhamento entre as camadas*



Fonte: *TANENBAUM, A.S. Redes de computadores, 4ª Edição, p.516, 615*

É perceptível que no *payload* de um quadro existe um pacote IP que, por sua vez em sua carga útil tem um TPDU (*Transport Protocol Data Unit),* que por sua vez em sua carga útil, tem um SPDU (*Session Protocol Data Unit*) e assim por diante.

Este aninhamento é importante para manter as funcionalidades entre as camadas isoladas e para que cada uma cumpra seu papel. O aninhamento entre camadas é ilustrado na **Figura 3**, e pode ser chamado também de modelo de boneca russa.

Enquanto o protocolo IP cuida da transmissão da informação entre remetente e destinatário, localizando os nós da rede em que o pacote deve ser entregue, os protocolos da camada de transporte cuidam da entrega de uma unidade de dados, os TPDUs que devem ser entregues para processos específicos para serem de alguma utilidade. A entrega dos segmentos aos processos corretos é feita por meio de dos números de portas. Dois protocolos são responsáveis por esta entrega, os protocolos TCP e UDP (COMER, 2000).

Uma **porta** é, simplificando, um número associado a um processo. Quando um segmento chega, ele é analisado para que se possa saber para qual processo ele será encaminhado, e isso é feito por meio de uma tabela de portas.

Existem números de portas amplamente conhecidos como 80 para o processo do servidor HTTP (*Hypertext Transfer Protocol*), 21 para o FTP (*File Transfer Protocol*) e outros (CASAD, 2004). Estes serviços serão abordados posteriormente.

O *Transport Control Protocol* (TCP – Protocolo de controle de transporte) tem como característica a garantia de conexão, gerada por meio de um processo de sincronia conhecido como *positive acknowledgement with retransmition* (reconhecimento positivo com

retransmissão) e implementado na forma do *three-way handshake* (traduzido livremente por triplo cumprimento) (COMER, 2000).

Com essa verificação o transmissor inicia a comunicação enviando no TPDU um campo denominado SYN, de sincronia, com um número sequencial. O receptor ao receber o pacote, envia uma resposta com um campo SYN com um valor sequencial próprio e com o campo ACK do transmissor incrementado em uma unidade. O emissor inicial recebe esta resposta e retorna ao destinatário uma nova confirmação, com seu campo ACK com o valor de SYN do destinatário incrementado. Com este processo, o protocolo estabelece uma sincronia que permite que, caso uma confirmação não chegue por muito tempo, um pacote seja reenviado e nenhuma informação seja perdida (TANENBAUM, 2003).

Além de prover a garantia de conexão, o protocolo TCP estabelece conexão fim a fim por meio dos pares IP/Porta do remetente e do destinatário. Essa característica dá uma vantagem extra ao TCP, já que permite que um processo ligado a uma porta possa se comunicar com mais de um nó de rede remoto (COMER, 2000). Entretanto, o protocolo TCP é dito como orientado à conexão, pois somente as máquinas (os elementos finais) estão cientes da conexão, mantendo variáveis de estado e reserva de memória para transferência de informações. Os roteadores intermediários não têm qualquer conhecimento da existência de uma conexão no caminho (KUROSE e ROSS, 2006).

Embora o protocolo TCP seja confiável, ele trás uma sobrecarga para garantir a sincronia e entrega dos pacotes. Esta sobrecarga pode não ser desejável em alguns cenários, como transmissão multimídia (processo conhecido por *streaming*), telefonia com voz sobre IP e vídeo conferência. Neste tipo de transmissão, onde a velocidade de transferência é mais importante que a perda de um ou outro pacote, um protocolo sem conexão é preferível (KUROSE e ROSS, 2006).

Para este tipo de situação, há um segundo protocolo de transmissão denominado *User Datagram Protocol* (UDP – Protocolo de datagrama de usuário).

O protocolo UDP, diferente do TCP, é um protocolo sem conexão e não prevê qualquer processo de *handshaking*. Quando o remetente deseja enviar algum tipo de informação, ele simplesmente o faz. Não há qualquer garantia, no entanto, da entrega do pacote e o remetente não fica sabendo da entrega do mesmo (KUROSE e ROSS, 2006).

Além dos protocolos TCP e UDP, há um terceiro protocolo, denominado *Internet Control Message Protocol* (ICMP) que é transportado na carga útil do protocolo pacote IP. O uso mais

comum para este protocolo é a notificação de erros. Quando um usuário navegando na Internet recebe uma mensagem de "rede de destino inalcançável", na verdade um roteador não conseguiu entregar a requisição e enviou um pacote ICMP com o erro ao remetente.

Assim como o UDP, o protocolo ICMP não tem garantia de entrega e se trata de um protocolo sem conexão (KUROSE e ROSS, 2006). Outra aplicação deste protocolo é com as aplicações *ping* e *traceroute*.

O *ping* emite uma requisição de eco ao destinatário que deve responder ao remetente, desse modo o mesmo é muito utilizado por técnicos em testes de conectividade em redes locais. Já o *traceroute* permite rastrear a rota que um pacote tipicamente percorre entre o remetente e o destinatário.

2.2.2. Roteamento

Uma rede é composta por elementos de borda, seus *hosts* finais, e por elementos intermediários, responsáveis por encaminhar os pacotes transmitidos entre os *hosts*. Um *host* é um computador, ou outro dispositivo capaz de se comunicar via rede, que hospeda as aplicações, arquivos ou serviços de rede.

Os elementos intermediários são os roteadores de núcleo, responsáveis pelo encaminhamento dos pacotes de um para outro até que atinjam seu destino. O roteamento é realizado por meio dos endereços de IP e pelas interfaces de rede. Quando um *host* envia um pacote, por exemplo, uma requisição de página, a um servidor, os roteadores deverão encaminhar essa requisição pelo caminho correto. Quando um pacote chega com um determinado destino, o roteador consulta sua tabela de roteamento e verifica por qual interface de rede deve enviar o pacote para que ele atinja seu destino (KUROSE e ROSS, 2006).

É possível configurar rotas estáticas e dinâmicas. Nas rotas estáticas o responsável pelo equipamento acrescenta as rotas manualmente na tabela de roteamento por meio da interface de configuração do roteador, ou via arquivos de configuração. No roteamento dinâmico, softwares específicos executam algoritmos específicos para calcular a rota até o destino (ANDERSON e BENEDETTI, 2009).

Há diversos protocolos de roteamento, cada um com suas qualidades, vantagens e desvantagens. O protocolo mais simples é o *Routing Information Protocol*[12] (RIP), sendo um dos mais difundidos e podendo ser encontrado em todos os roteadores do mercado.

Neste protocolo, a cada intervalo de tempo, o roteador envia sua tabela de roteamento para todos os seus vizinhos. Seus vizinhos recebem essa tabela e comparam com as suas fazendo as modificações necessárias nas próprias tabelas. Seus softwares de controle mantém o estado da rota durante algum tempo. Se o roteador que tem aquela rota não enviar uma atualização ele passa a acreditar que aquele roteador está *off-line,* ou seja, sem comunicação e recomeça a atualização.

Um dos pontos fracos do protocolo RIP é que ele usa apenas a quantidade de *hops* (saltos) como contagem para escolher a melhor rota quando duas possibilidades lhes são apresentadas. Comparando uma rede de computadores com um grafo não ponderado, *hop* é a quantidade de arcos ou arestas que são necessários para atingir outro nó da rede. Outros problemas deste protocolo são a baixa velocidade de convergência e o congestionamento gerado, já que ele emite continuamente pacotes com suas tabelas de roteamento.

Finalmente, a primeira versão deste protocolo não possuía qualquer dispositivo de segurança o que poderia permitir um roteador com problemas colocar rotas defeituosas na rede. Na segunda versão do protocolo, foi acrescentado um controle com senha para evitar a inclusão de rotas ruins (ANDERSON e BENEDETTI, 2009).

Além do protocolo RIP, outros protocolos de rotamento incluem o *Open Shortest Path First* [13](OSPF), O *Enhanced Interior Gateway Routing Protocol*[14](EIGRP) e o *Border Gateway Protocol*[15] (BGP).

O OSPF é um protocolo muito eficiente, pois permite o ajuste de várias métricas para calcular a melhor rota. Ele também requer autenticação e não confia em informações geradas por roteadores desconhecidos. Seu algoritmo de conversão é rápido e é suportado por vários fabricantes de roteadores. Entretanto, por possuir diversas métricas ele apresenta uma complexidade maior de configuração e consome maior quantidade de memória do roteador.

Embora utilize um algoritmo diferente, o EIGRP da fabricante Cisco tem as mesmas qualidades e pontos fracos do OSPF, exceto que, só é compatível com seus próprios

[12] Protocolo de informações de roteamento

[13] Primeiro o caminho mais curto aberto
[14] Protocolo avançado de roteamento de *gateway* interior
[15] Protocolo de *gateway* de borda

equipamentos. O EIGRP envia um pacote *Hello* (Olá) para seus vizinhos, se os vizinhos estão na mesma sub-rede e possuem o mesmo *Autonomous System Number*[16] (ASN) o vizinho envia sua tabela de roteamento para o remetente do *Hello*, que confirmará o recebimento com um pacote de conhecimento (ACK). O EIGRP usa o algoritmo de atualização por difusão (*Diffusing Update Algorithm* – DUAL) que evita a possibilidade de uma rota estabelecer um ciclo. Além disso, ele tem um mecanismo denominado *Reliable Transport Protocol* (RTP) que assegura que as informações enviadas não contém erros e o EIGRP só envia suas tabelas de roteamento quando há mudanças (ANDERSON e BENEDETTI, 2009).

Uma última questão será levantada referente ao protocolo IP e ao roteamento.

Devido ao crescimento da quantidade de dispositivos ligados à Internet, a quantidade de números de IPs disponíveis ficou escassa. Na realidade já teria se esgotado se todo e qualquer dispositivo conectado a Internet requeresse um IP único válido.

Um ISP recebe dos órgãos regulamentadores uma faixa de IPs que podem ser oferecidos aos seus clientes e utilizados nos próprios servidores. Ao considerar que um ISP recebe, por exemplo, um endereço na forma 200.100.0.0/16, tem-se que este ISP é capaz de atender pouco mais de 65.000 clientes. Em uma empresa com dezenas de computadores que precisam ser interligados à Internet seria inviável que o provedor fornecesse um IP para cada máquina interna. Deste modo, foi necessário encontrar uma alternativa, que veio na forma do NAT (*Network Address Translation*).

O NAT é um recurso dos roteadores que permite que dentro de uma empresa, ou mesmo residência, utilize-se uma faixa de IP privado para interligar seus computadores. Deste modo, uma empresa ou residência pode ter um único IP real ou externo, enquanto o NAT se encarrega do roteamento dos pacotes para a rede interna.

O NAT usa uma tabela de portas e manipula os pacotes IP, TCP e UDP trocando os endereços internos pelo externo no momento de enviar o pacote e, atribuindo ao campo porta de origem do pacote TCP ou UDP sua porta de saída livre. Ao destinatário é como se o roteador em questão fosse o remetente. Quando a resposta é devolvida, o roteador troca e reconstrói o pacote, colocando como destino o IP interno e as portas corretas e então encaminha para a rede interna o pacote (TANENBAUM, 2003).

Neste trabalho, é incluído o NAT na seção de roteamento, pois, embora seja efetuada de uma forma diferente, acaba sendo um meio de rotear o pacote ao seu destino correto dentro de

[16] Número do sistema autônomo, que é um identificador da rede

uma rede privada.

2.2.3. Serviços e aplicações

Até o momento este trabalho tratou das camadas inferiores, descrevendo como a rede dá suporte às aplicações que dependem essencialmente dela. A partir deste ponto, serão descritos os serviços e aplicações de alto nível, ou de suporte a estes.

Como foi descrito, os pacotes transitam entre os *hosts* usando os diversos protocolos de conexão e roteamento para atingir seu destino e que, o roteamento é feito por meio dos números IP e dos algoritmos de roteamento.

Entretanto, números são maneiras ineficientes para o usuário final localizar uma informação em uma máquina remota. Se a empresa que tiver a informação mudar de cidade, seu IP provavelmente será modificado e todos os seus clientes precisariam ser notificados da mudança.

Essa notificação poderia demorar o suficiente para que as empresas perdessem negócios, fora a possibilidade de um concorrente apropriar-se de seu IP anterior. Entrementes, como os IPs não categorizam as informações, um dígito incorreto poderia enviar uma informação para um local inesperado e indesejado.

Para isso, existem os nomes de domínio. Nome de domínio é um nome que pode ser registrado em uma empresa denominada "registrar" e então registrado em um *Domain Name Server* (DNS).

O DNS é um servidor de nomes que mantém um banco de dados que relaciona nomes de domínios com os IPs, de forma similar a feita pelo protocolo ARP na tradução de endereço físico para endereço IP. O serviço de DNS retorna o IP para um determinado nome quando uma consulta é feita pelo cliente DNS (TANENBAUM, 2003) e (ANDERSON e BENEDETTI, 2009).

Não basta que um *host* seja localizado, este deve prover uma aplicação para ter alguma utilidade na rede. Uma destas possíveis aplicações é o *servidor web* que é um software instalado em um *host* que hospeda páginas de Internet de empresas e outras instituições.

Um usuário final, por meio de um *browser*, o cliente *web*, faz uma requisição via rede a este serviço, que retorna a página desejada. Em nível de aplicação, o protocolo utilizado nesta comunicação é o protocolo *Hiper Text Transfer Protocol* (HTTP). O usuário digita a *Uniform Resource Locator* (URL) que é composta do nome do computador mais o domínio (FQDN –

Fully Qualified Domain Name) no navegador de Internet, e com isto será feita uma requisição ao servidor de DNS, retornando o IP daquele servidor. Em seguida a requisição HTTP será enviada ao servidor que responderá enviando o conteúdo da página requisitada ao cliente. Este servidor *web* poderá realizar qualquer que seja o processamento para a entrega desta página, montando-a dinamicamente ou simplesmente enviando uma cópia do arquivo requisitado.

No mercado os principais servidores *web* são o *Apache Web Server* da *Apache Software Foundation* e o *Internet Information Service* (IIS) da *Microsoft*.

Outro serviço muito comum da Internet é o serviço de *e-mail*, especificado pela RFC822[17], que determina a organização e o formato das mensagens, organizando os campos necessários para o envio e transmissão.

O correio eletrônico surgiu em 1982 e se tornou mais comum que as correspondências de papel. Os agentes de *e-mail* utilizam DNS Reverso para descobrir se o servidor de *e-mail* é realmente o servidor que gerencia as contas de um determinado domínio. Além da RFC822, outra especificação importante relacionada ao *e-mail* é a RFC13[18]41 que define o *Multipurpose Internet Mail Extension* (MIME). O MIME acrescenta cabeçalhos especiais para permitir o envio de *e-mails* em outros formatos que não caracteres ASCII puros. Além das especificações, há 3 protocolos que constituem o sistema típico de *e-mail*: o protocolo ***Simple Mail Transfer Protocol*** (SMTP), responsável pelo envio do *e-mail* do remetente ao destinatário. O software cliente do remetente envia a mensagem para o servidor de *e-mail*, também conhecido por *Mail Transfer Agent* (MTA – Agente de transferência de e-mail), que se encarregará de localizar o destinatário e realizar a entrega. Quando o destinatário deseja receber o *e-mail*, pode se valer de dois protocolos para a leitura; O ***Post Office Protocol*** (POP), mais simples, permite que o usuário por meio de seu cliente de *e-mail* transfira as mensagens do servidor para seu computador pessoal. Embora alguns clientes programem alternativas, o objetivo do POP é transferir as mensagens eliminando-os do servidor; e por fim o ***Internet Message Access Protocol*** (IMAP), com mais comandos, que permite que as mensagens sejam visualizadas remotamente, mas armazenadas em caixas de entrada no servidor (TANENBAUM, 2003).

Atualmente, no entanto, é comum o uso de um cliente *web* para visualizar as imagens, dispensando o software cliente no próprio computador, de modo a acessar a própria caixa de várias fontes.

[17] http://tools.ietf.org/html/rfc822 - acessado em 13/08/2014
[18] http://tools.ietf.org/html/rfc1341 - acessado em 13/08/2014

2.2.4. Gerenciamento de redes

Quando a rede é complexa, descobrir seus problemas não é uma tarefa trivial. Portas podem queimar, cabos podem romper, outros problemas podem aparecer de acordo com a variação da energia elétrica ou simplesmente por problemas de projeto, ocorrendo congestionamento em pontos críticos.

Há meios de detectar cada um destes problemas, acessando via SSH (*Secure Shell*) ou *telnet* os dispositivos, se for possível a comunicação. Algumas vezes é necessário ligar um computador fisicamente ao dispositivo para analisar por meio de uma porta serial. Para evitar a necessidade do deslocamento, existem sistemas de gerenciamento de rede que tornam esta tarefa mais eficiente (ANDERSON e BENEDETTI, 2009).

A ISO criou um modelo de gerenciamento agrupando em cinco grandes áreas (KUROSE e ROSS, 2006):

1. **Gerenciamento de desempenho**: Quantificar, medir e informar o desempenho mensurável dos elementos da rede.
2. **Gerenciamento de falhas**: Registrar, detectar e reagir às condições de falha nos elementos da rede.
3. **Gerenciamento de configuração**: Permite que o administrador saiba quais dispositivos fazem parte da rede administrada e quais suas configurações de hardware e software.
4. **Gerenciamento de contabilização**: Permite que o gerente de redes especifique, registre e controle o acesso dos usuários aos recursos da rede no que tange à quantidade de recursos que eles podem utilizar como quotas em disco, cobrança por transferência.
5. **Gerenciamento de segurança**: Permite ao administrador controlar as políticas de acesso aos dispositivos e informações que circulam pela rede.

Uma das formas eficientes de gerenciar o estado da rede é por meio de um protocolo denominado *Simple Network Management Protocol* (SNMP).

Cada dispositivo executa um software denominado agente SNMP que tem acesso a uma base de informações denominada *Management Information Base* (MIB). Essa base de dados é padronizada pela RFC1213, mas pode conter informações customizadas pelo fabricante.

Cada informação desta base é identificada por um *object identifier* (OID). O programa gerente SNMP acessa as informações das MIBs dos agentes e oferece ao administrador relatórios sobre o estado da rede.

Além do SNMP, o gerente de rede pode se valer de outro recurso, o *syslogd* (*daemon* de *log* de sistema), que pode ser configurado para enviar para o gerente as informações de *log* (auditoria) de acordo com o nível de relevância. Ao receber os *logs*, um programa pode verificá-lo e caso encontre algum registro que deva ser observado, pode enviar uma notificação ao administrador (ANDERSON e BENEDETTI, 2009).

2.3. Redes móveis

2.3.1. Comunicação celular e dispositivos móveis

Além das redes tradicionais discutidas na seção 2.2, o avanço da tecnologia permitiu o desenvolvimento de uma série de dispositivos capazes de romper as limitações tecnológicas provocadas pelo cabeamento, levando a criação das redes sem fio.

Uma vez que a rede sem fio passou a ser utilizada para a transmissão de dados, foi uma questão de evolução técnica e do *downsizing*[19] permitir não só que a conexão seja feita sem os fios mas, permitir que estes dispositivos pudessem ser deslocados à medida que o usuário se locomovesse. Desta forma, por meio de um telefone celular, *tablet*, ou outro dispositivo de acesso o usuário pode manter sua conexão com a Internet.

Assim, é possível perceber que existem vários usos para uma rede sem fio como: (1) a comunicação em telefonia celular; (2) comunicação entre veículos em movimento; (3) instalação de redes em locais onde não é viável realizar a instalação de uma fiação de transmissão de dados; (4) ou mesmo uso de sistemas de informação miniaturizados, para a realização de vendas ou pedidos, isso entre tantos outros exemplos (TANEMBAUM, 2003).

Neste ponto é importante que uma rede móvel seja bem definida. Kurose (2006) descreve uma rede móvel, em sentido amplo, como sendo uma rede onde um nó precisa trocar de ponto de acesso enquanto se desloca.

De fato, três níveis de mobilidade são definidos: em uma rede onde o usuário anda com seu notebook entre seu quarto e sua sala, conectado ao mesmo ponto de acesso, é possível definir como uma rede sem mobilidade; um estágio intermediário é obtido quando o mesmo notebook é desligado e conectado a Internet por meio de outro ponto de acesso, por exemplo, considerando a residência e o local de trabalho e; o ponto de máxima mobilidade é obtido quando a transição entre os dois locais é transparente, sem interrupção da conexão (KUROSE,

[19] Redução do tamanho do equipamento, miniaturização dos dispositivos.

2006). Este processo, conhecido como *handover* (KUROSE e ROSS, 2013) é importante para a mobilidade. Na **Figura 4** são ilustrados os diferentes graus de mobilidade.

Figura 4: *Graus de mobilidade*

Fonte: *Adaptado de Kurose e Ross (2013), Computer Networking: A top down approach, 6th edition, p.555*

Diferentemente de uma rede cabeada que leva o sinal ao ponto de destino considerando apenas a distância e força do sinal, que pode ser ampliada com o uso de repetidores, para a transmissão de dados sem fio é necessário utilizar antenas.

As antenas possuem características como o modelo de radiação. Uma antena isotrópica, ou omnidirecional, transmite e recebe o sinal de modo uniforme em todas as direções, enquanto uma antena direcional concentra o sinal em regiões específicas. Também é considerado neste caso o tipo de propagação, quer ela acompanhe a curvatura da terra, quer seja refletida pela ionosfera ou que os pontos possuam linha de visada, ou seja, não estejam presentes obstáculos entre antenas transmissoras e receptoras (STALLINGS, 2005).

Enquanto nas redes cabeadas é possível criar redes ponto a ponto por meio do uso de dispositivos como *switches* (KUROSE e ROSS, 2013), nas redes sem fio a comunicação é feita por difusão. Para minimizar os problemas ocorridos com a colisão de informações, representadas por interferência nos sinais, é necessário realizar um controle sobre o sinal transmitido. A partir do momento que se têm a presença de três ou mais nós que precisam se comunicar, o espaço de comunicação precisa ser compartilhado. Neste caso, o compartilhamento pode ser efetuado, seja utilizando protocolos que dividem o tempo de transmissão para cada nó (TDM - *Time-division multiplexing*), seja por meio da definição de faixas de frequência (FDM - *Frequency Division Multiplexing*), permitindo a comunicação entre os elementos (GARG, 2007).

Além da multiplexação para canais diferentes, dispositivos que estejam compartilhando um mesmo canal de transmissão precisam utilizar um protocolo de acesso ao meio. Nas redes cabeadas o protocolo é o CSMA/CD, pois é relativamente simples, via hardware, descobrir a

ocorrência de colisões. No caso das redes sem fio é necessário utilizar o protocolo CSMA/CA (*Collision Avoidance*, Prevenção de Colisão).

Quadros RTS (*Request to Send*, Requisição para enviar) e CTS (*Clear to sent*, livre para enviar) são enviados pelos nós com a finalidade de negociar um determinado período de tempo, fazendo com que os demais nós permaneçam apenas em estado de escuta sem realizar a transmissão. Quando os quadros RTS e CTS são negociados com sucesso, o quadro de dados é então enviado. Os quadros RTS possuem informação do tempo de duração requisitado e quando ocorrem colisões impedindo que estes quadros de controle possam ser lidos com sucesso, a espera aleatória é efetuada por todos os participantes da comunicação (WESOLOWSKI, 2002).

Os primeiros celulares desenvolvidos possuíam tecnologia de transmissão analógica, com compartilhamento por divisão de frequência. Para aumentar o número de dispositivos, as frequências eram distribuídas de acordo com células regionais. A tecnologia exata empregada, no entanto, poderia variar de acordo com a região.

A geração de celulares digitais iniciou com o CDMA (*Code Division Multiple Access*), que se utiliza de uma frequência alta para codificar os dados que serão transmitidos pela onda portadora. Novamente, esta geração não possui padronização internacional e outros protocolos e tecnologias foram criados concomitantemente (TANENBAUM, 2003).

A geração seguinte, denominada de 2G, utilizava a tecnologia GSM (*Global System for Mobile Communications*), que embora também trabalhe com compartilhamento por tempo e frequência, possui canais de comunicação mais largos, o que permite maiores taxas de transmissão.

A maior taxa de transmissão possibilitou a criação do GPRS (*General packet radio service*) que utilizava alguns dos *slots* de tempo do canal para a transmissão de pacotes de dados. Posteriormente, a geração de celulares 3G, com melhor eficiência espectral permitiu alcançar maiores distâncias e taxas de transmissão e maior potencial de comunicação de pacotes via Internet (STALLINGS, 2005).

A mobilidade também causa conflito no endereçamento IP do dispositivo móvel, uma vez que as rotas são diferentes e o endereço de IP do dispositivo pode não pertencer à rede da nova estação base.

Neste caso, é utilizado um *Care-of Address* (CoA), um endereço temporário cedido pela rede do agente base para permitir que o dispositivo seja localizado. Na sua forma mais simples, o dispositivo móvel informa à sua estação principal seu CoA, que redireciona os pacotes que

cheguem até ela para a estação base a qual o dispositivo móvel esteja conectado (STALLINGS, 2005).

Em seu trabalho, Santana (2003) mostra os problemas inerentes do protocolo TCP em relação as redes sem fio, pela impossibilidade de realizar a distinção, via TCP, de situações de congestionamento e queda da comunicação.

Diversas propostas de otimização foram desenvolvidas, como o protocolo *Snoop*, *Freeze TCP*, e TCP com reconhecimento seletivo. Basicamente, as propostas de otimização desta comunicação estabelecem a participação do ponto de acesso como elemento da comunicação e controle por meio de dos pacotes de confirmação ACK (SANTANA, 2003).

2.3.2. Redes sem fio

Além da tecnologia de comunicação de dispositivos móveis como celular, utilizando sinais GSM, 3G ou 4G, outra tecnologia de acesso é composta das redes locais sem fio, conhecidas como redes *Wireless* ou Redes *Wi-Fi*.

Esta rede é diferente da rede móvel descrita inicialmente, pois foi concebida como alternativa para interconectar computadores onde o cabeamento não puder ser realizado, em vez de prover acesso móvel.

As redes sem fio são baseadas nos protocolos de comunicação de rede 802.11, padronizados pelo LMSC[20] (IEEE 802 LAN/MAN *Standards Comitee, Institute of Electrical and Electronics Engineeers, 802 Local Area Network and Metropolitan Area Network Standards Comitee*, também conhecido como LMSC – Comitê de redes locais e redes metropolitanas 802 da IEEE). O LMSC define vários padrões de comunicação em redes, entre eles o 802.11 para redes sem fio.

Os dispositivos de uma rede sem fio podem se conectar com um elemento central ou estação base denominado *Access Point* (Ponto de acesso), ou então formar uma rede entre os pontos sem a presença deste nó central, denominada rede *Ad Hoc*[21].

O padrão 802.11 define a camada de acesso ao meio (MAC) e a forma como serão trocados os pacotes, enquanto quatro camadas físicas são definidas, pelos padrões 802.11a,

[20] http://grouper.ieee.org/groups/802/802%20overview.pdf – acessado em 01/09/2014
[21] Em latim, *ad hoc* significa para este propósito, o que combina com a configuração de uma rede deste tipo.

802.11b, 802.11g e 802.11n[22]. As diferenças mais sensíveis entre os padrões da camada física são: a taxa de transmissão, a quantidade de canais que podem se sobrepuser e a frequência de transmissão. (STALLINGS, 2005). Na **Tabela 3**, adaptada de Stallings (2005), são ilustradas as diferenças entre os padrões.

A linha canais ilustra quantos canais simultâneos podem ser utilizados em um mesmo ambiente sem sobreposição, enquanto a largura de banda ilustra a faixa de frequência utilizada por estes canais. A linha frequência, descreve a frequência fundamental da onda portadora do sinal, embora a frequência real de uso seja variável de acordo com os canais utilizados. A taxa máxima de transmissão ilustra o limite superior, embora as mesmas possam operar com taxas menores quando os nós estão mais distantes e a potência dos transmissores envolvidos não é suficiente para garantir a qualidade do sinal.

Tabela 3 : Caracteristicas do padrão 802.11

Padrão	802.11	802.11a	802.11b	802.11g	802.11n
Canais	3	4	3	3	4
Frequência (GHz)	2,4	5	2,4	2,4	2,4 ou 5
Máxima taxa de transm. (Mbps)	1,2	54	11	54	300 [23]
Largura de banda (MHz)	20	20	20	20	20 ou 40

Para identificar a rede sem fio a qual o nó está conectado, um quadro especial, denominado *Beacon*, é enviado pelo ponto de acesso. O *Beacon* tem a função de carregar o SSID (*Service Set IDentifier*), identificador de conjunto de serviço, e outras informações relacionadas ao tipo de características que a rede possui e aos recursos oferecidos. Alguns exemplos de informações são: qual o protocolo de segurança que será utilizado; rede com infraestrutura ou *Ad hoc*; taxas de transmissão suportadas entre outras informações (CARPENTER, 2008).

Como a comunicação sem fio utiliza uma rede por difusão sem prevenção de acesso à mídia de transmissão, é possível para qualquer usuário mal-intencionado capturar pacotes que estejam sendo transmitidos no momento.

[22] http://standards.ieee.org/getieee802/download/802.11n-2009.pdf – acessado em 02/09/2014

[23] Atingido por meio do uso de 2 canais de 40 MHz, o que pode ocupar boa parte do espectro de outros dispositivos no ambiente.

Na impossibilidade de proteger a mídia, o conteúdo dos pacotes precisa ser criptografado. Entre os protocolos de segurança de transmissão de dados sem fio, estão os protocolos WEP (*Wired Equivalent Privacy*, privacidade equivalente a cabeada), WPA (*Wi-Fi Protected Access*, Acesso protegido em rede sem fio) e WPA2 (*Wi-Fi Protected Access* 2, Acesso Protegido em rede sem fio versão 2), que implementam diferentes algoritmos de criptografia síncronos para realizar a proteção dos dados transmitidos.

Além da criptografia, protocolos como o WPA2 permitem o controle de autenticação, por meio do protocolo EAP (*Extensible Authentication Protocol*, protocolo de autenticação extensível) e suas derivações, incluindo autenticação por meio de chaves digitais e senhas que identificam os usuários unicamente (STALLINGS, 2005; CARPENTER, 2008).

2.4. Trabalhos relacionados

Nesta seção são descritos alguns trabalhos relacionados ao projeto com foco em redes sem fio, demonstrando alguns dos desafios atuais nestes tipos de rede.

Uma das preocupações recorrentes com redes móveis e sem fio é a questão da segurança e do transporte seguro e confiável dos dados entre os nós da rede. Uma das técnicas de transmitir dados de forma secreta é por meio da esteganografia que consiste em esconder os dados ou mensagens que deseja ser transmitida de forma oculta em outro arquivo aparentemente não relacionado.

O trabalho de Mukerjee e De (2012) mostra um exemplo de esteganografia para a transmissão de dados sigilosos por meio de dados criptografados com chaves simétricas e transmitidas inseridas no áudio de uma transmissão. Neste trabalho, os *bits* de dados são alocados por meio de uma série Fibonacci. O artigo ainda apresenta que a transmissão da mensagem pode ser feita com 75% do tempo que as abordagens anteriores utilizavam (MUKERJEE e DE, 2012).

Becker et. al. (2012) descrevem uma taxonomia de serviços para redes móveis. Seu trabalho trata da evolução dos dispositivos móveis, onde os mesmos não são mais utilizados apenas para serviços de voz e mensagens, mas para toda uma gama de aplicações distribuídas com liberdade similar às aplicações dos computadores pessoais (PCs), de modo que os dispositivos baseados no *Android* da Google e no iOS da Apple mudaram a forma como os celulares interagem com a Internet e como esses serviços são ofertados a seus clientes, alterando, inclusive, os serviços tradicionais de voz e mensagem por sistemas de VOiP como o

Skype e as mensagens no padrão SMS (*Short Message Service*, Serviço de Mensagem Curta) pelos tradicionais *e-mails* (BECKER et. al, 2012). Esse trabalho demonstra o crescimento e mudança de paradigma das redes móveis, importância refletida na evolução das próprias redes, nos aspectos de infraestrutura e segurança os quais este trabalho se contextualiza.

Malik, Malik e Mupta (2012) desenvolveram uma proposta de algoritmo de roteamento para redes *Ad hoc* móveis, as denominadas MANETs (*Mobile Ad Hoc Networks*). Uma preocupação deste trabalho é o distanciamento entre os nós que precisam realizar a comunicação, sendo que os pacotes podem ser entregues com maior eficiência caso exista o roteamento entre os nós que compõe a rede *ad hoc*. Esse projeto trata de um trabalho em andamento, cuja hipótese central é melhorar o desempenho da comunicação e cujo funcionamento poderia ser construído e validado por meio de um protótipo desenvolvido com a ferramenta proposta neste trabalho.

O trabalho de Rao, Naidu e Seetharam (2012) propõe um agente inteligente por meio de lógica "fuzzy" para aperfeiçoar o sistema de localização dos telefones GSM. A localização dos dispositivos é um conceito importante para as operadoras de telefonia pois interfere na tarifação e precisa ser realizada independentemente da existência de outras tecnologias, como o GPS. No sentido desse trabalho, a localização se refere não as coordenadas espaciais do dispositivo, mas a quais células de transmissão o dispositivo está conectado durante as chamadas. Novamente, um protótipo funcional poderia ser construído por meio da utilização da ferramenta proposta neste trabalho.

Aoyama (2009) descreve as tecnologias fundamentais implementadas para a Internet e para a próxima geração de redes (NGN, *Next Generation Network*), ambas baseadas em endereços IPs e os problemas resolvidos por tais pesquisas, como a implementação de QoS e realizadas por meio de VoIP. Entretanto, esse trabalho demonstra alguns problemas inerentes as arquiteturas adotadas, criação de novos protocolos e pacotes sobre tecnologias existentes gerando sobrecargas e propondo problemas de pesquisa para a denominada NWNG (*New Generation Network*, Rede de Nova Geração). Um dos fatores abordados pelo artigo, que trata de desafios a serem atingidos até o ano de 2020, é a plataforma de testes das novas arquiteturas. O objetivo é viabilizar a entrega de novos conteúdos, como o cinema digitalizado a 4K e o que é denominado de ODS (*Other Digital Stuff*) que requererá maiores bandas de transmissão na faixa dos gigabits (AOYAMA, 2009). Aqui mais uma vez a plataforma desenvolvida supriria as necessidades de uma plataforma de teste para novas tecnologias/arquiteturas.

Outro desafio em redes móveis é descrito por Liu e Xu (2012), onde é abordado o problema da qualidade de serviço mediante a heterogeneidade de aplicações e serviços disponíveis, descrevendo as soluções de qualidade de serviço existentes como não preparadas para o tráfego de tais informações, uma vez que elas não tem informações sobre a carga de tais aplicações e serviços. Diante desse fato, uma arquitetura de serviços aberta para controle de qualidade é proposta e um estudo de caso é demonstrado, com a implementação da arquitetura permitindo a visualização de vídeo em alta qualidade (LIU e XU, 2012). Provendo um plataforma de virtualização de redes móveis sem fio é possível efetuar quaisquer tipos de análises e avaliações, inclusive de QoS de forma mais eficiente.

2.5. Considerações Finais

Neste capítulo foram apresentados os conceitos básicos de redes de computadores e redes móveis e sem fio. É possível observar que o uso do ar como meio de comunicação e difusão torna um pouco mais trabalhosa a localização dos dispositivos de modo que a mensagem a ser transmitida seja entregue ao destinatário correto. Outro problema é a possibilidade de captura dos pacotes que são trafegados que prova a existência de maiores exigências de segurança aos dispositivos. Desse modo, o próximo capítulo apresenta os conceitos básicos de georreferenciamento, o que permite um entendimento melhor dos problemas relacionados a movimentação dos elementos em uma rede móvel.

Capítulo

3

3. Georreferenciamento

3.1. Considerações Iniciais

Este capítulo descreve o funcionamento do sistema de georreferenciamento por meio de um sistema global de navegação por satélites, denominado GNSS (*Global Navigation Satellite System*). Embora a área de georreferenciamento extrapole facilmente o assunto aqui descrito, o georreferenciamento assistido por GNSS é de interesse direto do presente trabalho, embora existam técnicas e medidas terrestres que utilizam instrumental para aferições de distâncias, medidas topográficas e informações geográficas que não fazem parte do escopo do presente trabalho.

Na seção 3,2 é realizada uma breve introdução à teoria dos erros, elemento necessário para a compreensão dos erros de precisão dos sistemas de navegação. A seção 3.3 descreve os sistemas de coordenadas utilizados por tais sistemas. A seção 3.4 faz uma descrição do funcionamento do sistema GPS (*Global Positioning System*). A seção 3.5 descreve outros sistemas de localização por satélite. A seção 3.6 descreve trabalhos correlatos a este plano de mestrado que poderiam se beneficiar da ferramenta proposta, enquanto a seção 3.7 apresenta as considerações finais e a ligação entre o assunto aqui descrito e o presente trabalho de mestrado.

3.2. Teoria dos erros

Para a compreensão de um sistema de navegação, é importante compreender inicialmente os problemas existentes com a precisão das informações e como estes problemas são gerenciados.

Considerando qualquer que seja a unidade de medida, o processo metrológico consiste em comparar uma unidade de grandeza aferida por meio da medição com outra de referência da mesma natureza, chamada unidade padrão. Considerando a medida de distância entre dois objetos, na unidade metro, a medição é realizada por meio de uma ferramenta, que tem sua qualidade inerente e que pode interferir contribuindo com algum erro de aferição. A habilidade do operador também pode interferir no resultado e finalmente, as condições do ambiente,

temperatura, pressão e umidade podem interferir na precisão dos instrumentos utilizados (RABINOVICH, 2005).

Uma grandeza é um atributo físico mensurável quantitativamente, que pode ser observado e reconhecido de modo qualitativo. Na **Tabela 4** são ilustradas algumas grandezas reconhecidas no sistema internacional de unidades e suas unidades de medida padrão (RAMALHO, NICOLAU e TOLERO, 2007).

Tabela 4: Grandezas e unidades do sistema internacional

Grandeza	Unidade
Distância	metro
Tempo	segundo
Massa	quilograma
Corrente elétrica	ampere
Diferença de potencial	volt
Temperatura	graus kelvin
Quantidade de matéria	mol

Duas observações consecutivas podem resultar em valores de medidas diferentes, de fato, um axioma da metrologia é a existência de uma diferença entre o valor obtido por meio de uma medida e o valor real da grandeza.

Entre duas medidas diferentes realizadas com instrumentos de boa qualidade, é possível existir uma pequena diferença, que recebe o nome de discrepância (RABINOVICH, 2005). Entre várias medidas, é comum aceitar o valor médio como sendo o valor mais próximo da medida real. Entretanto, algumas vezes uma das medidas pode apresentar um erro grosseiro que interfere na média. Este valor anômalo, conhecido pelo termo em inglês *outlier*, pode ter origem na falha do instrumento de medição, no engano do operador ou em mudanças bruscas que possam interferir no processo externo (WEISBERG, 1985; RABINOVICH, 2005).

Considerando a existência dos erros de medida, faz-se necessário aplicar algum método para avaliar a existência dos mesmos de modo que uma medida mais precisa possa ser efetuada. Deste modo, *outliers* devem ser detectados e verificados para a correta detecção dos valores médios e mais aproximados.

3.3. Sistema de coordenadas

É comum que receptores domésticos façam a localização do receptor em mapas em suas respectivas telas, de modo que o usuário possa compreender rapidamente sua localização. Entretanto, para compreender como essa localização é efetuada é necessário entender que os satélites estão se locomovendo na abóboda celeste e que a localização é o resultado de um sistema de equações envolvendo dados do receptor e alguns destes satélites.

Para descrever essa localização é necessário compreender um sistema de coordenadas adequado, capaz de medir a posição e o tempo da determinada leitura.

Para compreender um sistema de coordenadas é necessário definir o que são coordenadas e o que elas representam. No primeiro momento tem-se uma coordenada de uma dimensão, utilizada para medir a distância entre um ponto qualquer e outro ponto de origem. A quilometragem de uma rodovia a partir de um marco origem é uma coordenada deste tipo.

Uma coordenada de duas dimensões é uma coordenada descrita como um par de valores, geralmente expressos na forma (x, y), definidas como as distâncias horizontais e verticais a partir do ponto de origem. Uma coordenada tridimensional considera não somente as distâncias da origem no plano, mas a distância do ponto ao plano bidimensional em linha perpendicular, valor conhecido como altura, e a representação desta coordenada é feita normalmente por meio da notação (x, y, z) (SEGANTINE, 2005)

No primeiro instante é importante localizar cada um dos satélites em relação a um referencial. Dado o centro da terra, chamado também de geocentro, como ponto de origem, o plano coincidente com a linha do equador e o plano coincidente com o meridiano de Greenwich, é possível estabelecer um sistema de coordenadas cartesiano de três dimensões. Tal sistema, descrito de forma geométrica, compõe um espaço chamado de espaço R^3 (XU, 2007).

A topografia, ciência que usa da Geografia e Geodésia para estudar as características da superfície da terra, e a cartografia, ciência que descreve os mapas de localização, utilizam vários sistemas diferentes em suas medidas.

O sistema topográfico usa medidas cartesianas onde os objetos da superfície de um terreno devem ser projetados verticalmente em um plano horizontal, estabelecido a uma determinada altitude. No Brasil, essa altitude é obtida por meio do *Datum Vertical* definido pelo *nível médio* do mar no porto de Imbituba em Santa Catarina.

O sistema planimétrico transforma as medidas de distância entre objetos de uma superfície projetados em um plano horizontal, com distâncias paralelas e norte orientado, geralmente, como o eixo Y. As coordenadas planimétricas são as mais utilizadas também para a criação de plantas baixas na engenharia civil e arquitetura (SEGANTINE, 2005).

A desvantagem destes sistemas, no entanto, é pressupor que a terra seja plana. Em pequenas regiões isso não apresenta problema, mas utilizando medidas de grandes distâncias, a curvatura da terra deve ser levada em consideração.

Desta forma, um sistema de coordenadas adaptado ao elipsoide que representa o planeta Terra foi criado e, as medidas de posição são descritas por meio de coordenadas de latitude, longitude e altitude relativa a este elipsoide, que no caso do sistema GPS (*Global Positionning System*) é chamado elipsoide WGS-84.

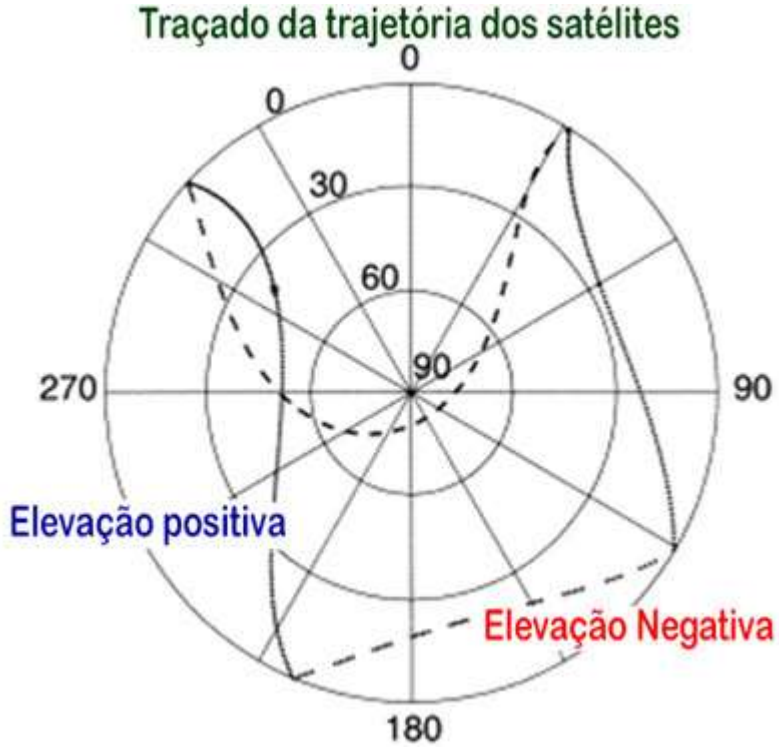
Este sistema recebe o nome de ECEF19 (*Earth Centered Earth Fixed*), pois as coordenadas são fixas independentes da rotação da terra, movimento que trás desafios quando se fala de objetos que estão em deslocamento (PRASSAD e RUGGIERI, 2005).

A longitude é o ângulo diedro formado pelos planos do Meridiano de Greenwich e do meridiano que passa pelo ponto considerado. A longitude pode ser contada no sentido oeste, quando é chamada Longitude Oeste de Greenwich (W Gr.) ou Negativa. Se contada no sentido Este, é chamada Longitude Este de Greenwich (E Gr.) ou Positiva.

A latitude é o ângulo formado pela normal à superfície adotada para a Terra, que passa pelo ponto considerado e a reta correspondente à sua projeção no Plano do Equador. A latitude quando medida no sentido do Polo Norte é chamada Latitude Norte ou Positiva. Quando medida no sentido do Pólo Sul é chamada Latitude Sul ou Negativa. Sua variação é representada indicando o sentido Norte ou Sul (O° a 9O°N / O° a 9O°S), ou por meio do sinal negativo para indicar a latitude do hemisfério inferior (O° a + 90° / O° a − 90°) (IBGE, 2013).

Neste sentido é interessante observar que, embora os satélites do sistema de navegação estejam em órbitas elípticas ao redor da terra, os mesmos não são geoestacionários. Deste modo, o movimento da terra produz como efeito um movimento aparentemente irregular na superfície da terra. Este efeito está ilustrado na **Figura 5** (KINTNER e LEDVINA, 2005).

Figura 5: *Movimento aparente de um satélite a partir do angulo de visão de um observador fixo. A linha sólida representa altitudes acima da linha do horizonte e a linha pontilhada, altitude inferior a linha do horizonte.*



Fonte: *Adaptado de Kintner e Ledvina (2005).*

3.4. Sistemas de navegação por satélite GPS

O GPS (*Global Positioning System*, sistema de posicionamento global) é um sistema americano criado pelo Departamento de Defesa dos Estados Unidos. Este sistema se tornou completamente operacional em 1994 e qualquer receptor GPS pode receber os sinais transmitidos por este sistema gratuitamente.

O sistema GPS é composto de três segmentos: um segmento espacial formado por 32 satélites distribuídos em 6 planos orbitais, inclinados a 55ª em relação ao eixo de rotação da terra a 21.000 Km de altura. Seu tempo de órbita é de 12 horas.

O segmento de controle é composto de estações terrestres que realizam a manutenção do relógio e das coordenadas dos satélites e seu monitoramento. A sede de controle fica em *Colorado Springs*, nos Estados Unidos, e as outras quatro estações de monitoramento são localizadas no Havaí, Ilha da Ascensão localizada no oceano Atlântico, Diego Garcia localizada no oceano Índico, e as estações do Havaí e Kwalajein no oceano Pacífico.

O último segmento, o de usuário, é composto pelos receptores de sinal GPS que possuem a função de calcular suas próprias coordenadas a partir do sinal recebido dos satélites (PRASAD e RUGGIERI, 2005).

Cinco faixas de frequência são alocadas para os transmissores e receptores GPS. Destas, as frequências L1, L2 e L5 podem ser utilizadas para aplicações civis, enquanto as demais, inclusive a L1 e a L2, transportam códigos criptografados específicos para aplicações militares.

A frequência fundamental, L1, é gerada a partir de um oscilador atômico de rubídio, na frequência de 12 MHz. Por meio de um circuito eletrônico multiplicador da ordem de 154 vezes, um sinal senoidal é gerado como portadora, transmitindo a 1575, 42 MHz. A fase do sinal é então modulada com 3 sinais diferentes. O sinal civil C/A de 1 MHz, o código militar P(Y) de 10 MHz e os dados de navegação de 50 Hz, ou seja, apenas 50 bits de informação por segundo são transmitidos, o que explica que receptores mais antigos precisem de algum tempo para realizar a primeira localização (PRASAD e RUGGIERI, 2005). Os dados transmitidos incluem a identificação do satélite, suas coordenadas e seu relógio interno.

A frequência L2, de 1227,60 MHz, transmite 3 códigos modulados, sendo o código P(Y) a 10 MHz, o código C (civil) a 1 MHz e um código M (militar) também a 1 MHz. Quando um receptor civil é capaz de receber as duas faixas de frequência L1 e L2, o uso do canal L2C permite a correção de atrasos de sinal gerados pela ionosfera melhorando a precisão das coordenadas obtidas. Em 2012, 2 satélites começaram a transmitir também a faixa L5C, na frequência de 1176,45 MHz. O sinal L5C foi desenvolvido para permitir o desenvolvimento do chamado transporte seguro de vidas (*safety-of-life*) e outras aplicações de alto desempenho. Esta banda é reservada especialmente para serviços de segurança em aplicações aviônicas (PRASAD e RUGGIERI, 2005; NCO, 2013).

A comunicação do satélite com o usuário é unidirecional. O satélite transmite os sinais nas frequências anteriormente descritas com os códigos modulados e, os receptores capazes de decodificar estes códigos nas frequências habilitadas recebem e decodificam estes sinais.

Os dados de navegação numéricos, denominados *bits,* e os códigos binários digitais, denominados *chips,* permitem identificar o satélite que transmitiu o sinal, suas coordenadas e o momento exato em que o sinal foi transmitido (XU, 2007).

Com as coordenadas e o momento exato em que o satélite transmitiu o sinal, a distância do satélite é calculada pela Equação 1, com d sendo a distância em metros, v a velocidade de

propagação do sinal e Δt a diferença do tempo entre recepção (tF) e transmissão (tI). A diferença de tempo é dada pela Equação 2.

$$d = v \times \Delta t \quad (1)$$
$$\Delta t = t_f - t_i \quad (2)$$

A distância, no entanto, pode ser calculada conhecendo as coordenadas de dois pontos, por meio da Equação 3.

$$d = \sqrt{(x_S - x_R)^2 + (y_S - y_R)^2 + (z_S - z_R)^2} \quad (3)$$

onde d é a distância entre os satélites, (xS, yS, zS) é a coordenada cartesiana do satélite e (xR, yR, zR) é a coordenada cartesiana do receptor. Obtendo o sinal de 3 satélites com (xS, yS, zS) conhecidos, é possível montar um sistema de 3 equações a 3 incógnitas com solução possível e determinada.

O problema então reside na precisão do valor de d, uma vez que a onda eletromagnética é transmitida ao valor da velocidade da luz de 299.792.458 m/s. A distância de um satélite posicionado exatamente na mesma latitude e longitude (xS = xR e yS = yR), ou seja, perpendicular ao plano tangente ao receptor é de aproximadamente 21.000 metros de altura, nos leva a um tempo de transmissão de 70,048 μs, ou seja, nesta perpendicular, 1 metro precisa de 3,3 ns para ser percorrido, logo, um erro de alguns poucos nano segundos é capaz de posicionar um elemento a vários metros de distância de sua posição real.

A sincronia do relógio do receptor na casa de nano segundos seria então, um desafio. Uma vez que as estações do segmento terrestre garantem que os relógios atômicos dos satélites estejam sincronizados e que apenas o tempo relativo é de interesse, uma solução é utilizar um quarto satélite e o sistema apresentado nas equações 4.

$$\begin{aligned} v \times (t_R - t_{S1}) &= \sqrt{(x_{S1} - x_R)^2 + (y_{S1} - y_R)^2 + (z_{S1} - z_R)^2} \\ v \times (t_R - t_{S2}) &= \sqrt{(x_{S2} - x_R)^2 + (y_{S2} - y_R)^2 + (z_{S2} - z_R)^2} \\ v \times (t_R - t_{S3}) &= \sqrt{(x_{S3} - x_R)^2 + (y_{S3} - y_R)^2 + (z_{S3} - z_R)^2} \\ v \times (t_R - t_{S4}) &= \sqrt{(x_{S4} - x_R)^2 + (y_{S4} - y_R)^2 + (z_{S4} - z_R)^2} \end{aligned} \quad (4)$$

Onde (xSn, ySn, zSn) é a coordenada cartesiana do satélite n, e (xR, yR, zR) são incógnitas e representam a coordenada do receptor. Adicionalmente, o conjunto {tS1..tS4} representam os

clocks internos dos satélites transmissores, que podem até mesmo ser diferentes neste caso, e tR o relógio interno do receptor descoberto por meio das equações. Novamente, v é a constante de velocidade da luz de aproximadamente 300 mil Km/s.

O cálculo do sistema de equações apresentadas fornece as coordenadas, porém erros de medição, efeito Doppler[24], reflexões e outras interferências físicas podem provocar erro na precisão da leitura do dispositivo. Alguns dispositivos utilizam técnicas diversas para reduzir o erro apresentado ao usuário. Uma destas técnicas é o filtro estendido de Kalman, que utiliza uma unidade inercial para aperfeiçoar a precisão do GPS no contexto de navegação, utilizado principalmente em veículos aéreos. No escopo deste trabalho, utilizamos somente o cálculo das coordenadas brutas (raw data), obtidos geralmente em dispositivos de GPS científicos.

3.5. Outros sistemas de navegação

Além do sistema americano, dois outros sistemas de navegação estão ativos no momento, o sistema *Glonass* de origem Russa e o *Galileo*, o sistema europeu. O sistema Glonass também é composto de uma constelação de 24 satélites e possui cobertura global, embora a quantidade de receptores comercializados mundialmente seja inferior a quantidade de receptores GPS.

As diferenças principais, além da compatibilidade com os receptores militares russos, é a melhor qualidade em altas latitudes onde o sinal do GPS civil pode não ser preciso. O sistema *Glonass* também teve problemas com lançamentos de satélites e somente em 2010 a constelação pode ser reativada (XU, 2007).

Já o sistema *Galileo* criado pelo programa espacial europeu e mantido pela união europeia é um sistema desenvolvido para atender as necessidades de localização independente dos sistemas GPS, *Glonass* e *Compass* (este último chinês).

As duas estações bases em terra são localizadas na Alemanha e na Itália. A constelação *Galileo* deverá ser composta por 30 satélites, sendo 3 reservas e 27 operacionais, que deverão estar em órbita em 2019. No momento, apenas 4 satélites estão operacionais para uma fase de validação e testes.

O propósito do sistema *Galileo* é apresentar precisão melhor que qualquer outro sistema, com erros de apenas 1 metro, seja na localização vertical ou na horizontal, enquanto os sistemas

[24] Efeito físico que modifica a frequência da emissão da onda, do ponto de vista do receptor, causado pelo deslocamento do emissor, que no caso de um satélite provoca variação de alguns quilohertz, dependendo da velocidade do satélite e da própria topografia do receptor/estação de base (EVANS e MCDICKEN, 2000).

Glonass e GPS têm aproximadamente, 5 metros na horizontal e 15 na vertical, considerando a faixa civil.

Uma função diferencial da rede *Galileo* é a capacidade de receber sinal de emergência dos receptores, informação que será repassada a um centro coordenador de resgates e confirmada então aos receptores, informando de sua situação. Existe a previsão de diferenciar os serviços de localização e resgate em modalidades gratuitas e pagas (PRASAD e RUGGIERI, 2005).

3.6. Trabalhos correlatos

Nesta seção são apresentados alguns trabalhos correlatos na área de georreferenciamento, em particular com o uso de receptores GPS, demonstrando as preocupações dos pesquisadores na área.

O trabalho de Yozevitch, Moshe e Levy (2012), demonstra um processo pelo qual é utilizada a reflexão dos sinais nas paredes dos altos prédios em áreas densamente urbanizadas, os chamados "*canyons urbanos*" podem ser utilizados para efetuar correções mais precisas no cálculo das coordenadas. Um grande diferencial deste trabalho é que, seja por meio de simulação ou de experimentos em campo, eles conseguem demonstrar que a precisão do GPS pôde ser reduzida para leituras com erro inferior a 1 metro, medidas que só conseguem ser realizadas por meio de recursos auxiliares. Isso vai ao encontro do que se pretende neste trabalho de mestrado que é a inclusão de formas de identificação da localização dos elementos, mesmo que na virtualização.

O trabalho de Merbitz et. al. (2012) é um exemplo de aplicação do uso de georreferenciamento e de sistemas de informação geográficas. Por meio de variáveis de calor e qualidade do ar em áreas urbanas, o sistema pretende prever as condições climáticas e de saúde nas regiões analisadas em longos períodos de tempo, por meio da comparação com outros locais em piores situações. A aplicação explorada por este cenário mostra um exemplo de cenário que poderia ser construído para validar o funcionamento da ferramenta proposta neste trabalho de mestrado, por meio da coleta de informações de uma rede de sensores virtuais montadas com os chamados dispositivos embarcados virtuais, estes discutidos posteriores.

Thorar et. al. (2012) apresentam um trabalho de coleta de informações geográficas por meio de redes sem fio móveis, onde computadores são ligados em rede por meio de telefones celulares. O trabalho mostra duas questões, uma relativa aos dados coletados e outra ao processamento distribuído dos dados obtidos. Novamente, o trabalho apresentado mostra um

estudo de caso onde a distribuição e processamento das informações poderia ser avaliado na ferramenta proposta neste trabalho, essencialmente pelo uso de um notebook para a coleta de informações.

O trabalho de Tmazirte et. al. (2012) mostra o uso de uma fusão de sensores, ou seja, um conjunto de sensores coletando diferentes formas de dados para gerar o mesmo tipo de informação, é utilizado para aperfeiçoar a integridade dos dados obtidos via GPS em redes veiculares. No cenário estudado por esse trabalho, os erros do cálculo das coordenadas GPS são críticos a funcionalidade e um conjunto de filtros precisam ser implementados. Este cenário poderá ser reproduzido neste trabalho de mestrado por meio da utilização dos sensores virtuais, uma vez que estes sejam estendidos para prover as informações requisitadas pelo sistema de filtros.

Gary e McGraw (2012) descrevem em seu trabalho um estudo sobre sistemas de "aumentação", utilizados para aumentar a precisão das informações obtidas pelos sistemas GPS, em especial utilizados na aviação. O foco do trabalho é o problema dos erros causados pela troposfera, indesejáveis e perigosos no contexto da aviação civil. O trabalho descreve como possibilidade, os sistemas de "aumentação" existentes não acompanharem a evolução dos recursos dos sistemas de localização, essencialmente por descartarem a correlação entre os sinais aumentados de duas fontes distintas. Neste caso, o sistema proposto neste trabalho poderia ser utilizado como plataforma para testar o funcionamento básico do software, por meio de casos de testes, antes de realizar testes em campo.

3.7. Considerações Finais

Este capítulo apresentou os principais conceitos relacionados com georreferenciamento, no sentido de localização de receptores por meio da rede de satélites. No próximo capítulo, será realizada uma revisão sobre os sistemas de virtualização, elemento central deste trabalho.

Capítulo

4

4. Virtualização de redes

4.1. Considerações Iniciais

Este capítulo descreve as características das máquinas virtuais e a virtualização de redes, seus fundamentos e o que diferença a virtualização das redes de outras ferramentas de simulação de redes, além de salientar suas vantagens para os propósitos deste trabalho de mestrado.

Inicialmente é feita uma introdução ao funcionamento de máquinas virtuais. Em seguida, são descritas as funcionalidades de uma ferramenta de virtualização de redes e sua comparação com ferramentas de simulação. É apresentada também a arquitetura UML (*User Mode Linux*), pois é um elemento central para a execução deste projeto. Finalmente, são descritos trabalhos de virtualização de redes e aplicações relevantes para o presente trabalho de mestrado.

4.2. Virtualização de sistemas operacionais

A virtualização de sistemas operacionais teve início com a separação do sistema operacional em camadas abstratas que permitiram o isolamento de processos.

O fundamento que viabiliza o funcionamento de tais máquinas é a possibilidade de abstrair todo o hardware para um determinado processo, onde este recebe uma cópia virtual do hardware. Este processo tem a "ilusão" de ter um processador próprio e seu próprio espaço de memória virtual. O hardware da máquina virtual é igual ao hardware do hospedeiro (SILBERCHATZ, GALVIN e GAGNE, 2009).

A ideia de virtualização de sistemas operacionais foi desenvolvida em 1972, quando a IBM, para compartilhar seus *mainframes* entre vários usuários criou o VM370. O VM370 dividia o *mainframe* em múltiplas máquinas virtuais, cada uma rodando uma instância separada se seu sistema operacional. O maior problema deste *mainframe* foi o compartilhamento das unidades de discos, resolvidos com a separação nos chamados *minidisks*, instâncias virtuais do sistema de arquivos.

O sistema operacional desta máquina, o CMS (*Conversional Monitor System*), capturava as chamadas de sistema na instância de máquina virtual, e só depois ela era encaminhada ao hardware. Nenhuma chamada de sistema acessava o hardware diretamente, o que fazia com que a instância do CMS realmente parecesse um computador real. As versões mais modernas deste *mainframe*, como o z/VM, é capaz de rodar diferentes sistemas operacionais, como o próprio *Linux*, em paralelo com o sistema operacional da IBM (TANEMBAUM, 2007).

Embora mais de 40 anos tenham se passado desde a existência dos primeiros sistemas virtuais da IBM, é recente a adoção das máquinas virtuais nas arquiteturas x86/64. A combinação de novas exigências de hospedagem de *sites*, servidores Apache, MySQL, aplicações de *Web Services* específicas, muitas vezes em servidores ou mesmo sistemas operacionais trouxeram novamente o assunto a tona.

Ao invés de oferecer inflexíveis hospedagens compartilhadas ou caras hospedagens de servidores dedicados, denominadas *co-location*, os clientes podem realizar a opção de contratar um serviço chamado *Virtual Private Server* (VPS, Servidor Privado Virtual). Nesta modalidade, o cliente consegue controle sobre a administração do sistema operacional e dos softwares instalados, e a empresa de hospedagem oferece recursos sobre demanda, como quantidade de disco, memória e tempo de processamento (SILBERCHATZ, GALVIN e GAGNE, 2009).

4.3. Taxonomia de sistemas virtualizados

Segundo Silberchatz, Galvin e Gagne (2009), existem três formas básicas de realizar a virtualização de sistemas operacionais. A virtualização completa, a simulação e a para-virtualização. Todas elas são consideradas formas de emulação de um sistema operacional (SILBERCHATZ, GALVIN e GAGNE, 2009).

- ✓ **Virtualização completa**: todo o hardware é virtualizado, de modo que o sistema operacional sendo executado em uma máquina virtual não recebeu nenhuma modificação para sua execução, ou seja, utilizando a expressão de Silverchatz (2009), o sistema operacional "acredita" que está sendo executado em um computador real.

- ✓ **Simulação/Emulação**: A simulação é a segunda forma de emulação de um sistema operacional ou de aplicações escritas para um sistema operacional, que ocorre quando um software escrito para uma plataforma/sistema operacional diferente será executado em uma máquina. Um software emulador será responsável por

interpretar as instruções da aplicação ou sistema operacional e traduzi-las para a nova arquitetura. Os dois desafios da simulação são a escrita de um processador completo, ou de todo um conjunto de chamadas de sistema, em software para interpretar as instruções do sistema emulado e o desempenho da aplicação, já que a tradução é custosa.

- ✓ **Para-virtualização**: Este tipo de virtualização ocorre quando o sistema executado nas máquinas virtuais precisa de modificações que viabilizem sua execução como máquinas virtuais, ou seja, ocorre uma adaptação do sistema para que o mesmo seja executado no ambiente virtualizado.

4.4. Virtualização de redes

A virtualização de redes leva a tecnologia de virtualização a um patamar diferenciado, onde não só as máquinas virtuais são criadas, como também os enlaces entre os nós podem ser controlados. Este tipo de ferramenta permite a criação de simuladores e cenários de experimentação que exigiriam vários recursos de hardware para serem executados (RIMONDINI, 2007).

Neste momento é importante separar as redes compostas de máquinas virtuais, a qual este trabalho se refere, das redes virtuais providas por meio da execução de várias redes simultâneas, cada uma com um propósito específico, ao mesmo tempo sobre um mesmo espaço compartilhado. Redes virtuais, como as descritas na página *Network Virtualization*[25] da Universidade de Princeton não fazem parte do escopo direto deste trabalho.

Entre as ferramentas de virtualização de redes no sentido utilizado neste trabalho, existe a ferramenta *Netkit*, as ferramentas *GINI*, *VNUml* e as ferramentas clássicas de virtualização de sistemas operacionais, *VirtualBox (ORACLE, 2014), Xen (XEN PROJECT, 2014), VMWare (VMWARE, 2014).*

[25] http://www.cs.princeton.edu/~jrex/virtual.html – acessado em 05/07/2014

4.4.1. A ferramenta Netkit

O software *Netkit* é um emulador de redes que permite a criação de experimentos de redes de computadores virtuais, incluindo os dispositivos de hardwares necessários para seu suporte como roteadores, servidores, *switches*, e da criação dos enlaces.

Além do hardware, estes equipamentos virtuais são inicializados com softwares reais que, em execução oferecem experiência real ao estudante para a realização de diversos estudos, mesmo que tenha apenas um computador em seu domicílio.

O *Netkit* utiliza softwares de código livre, principalmente licenciados pela GPL (*General Public License*), usando em suas máquinas virtuais o *Kernel* UML.

Para montar uma rede o *Netkit* usa um conjunto de arquivos de configurações e pastas, que formam um laboratório virtual. Um laboratório também pode ser inicializado por meio de *scripts* ou por meio da linguagem *NetML* que é uma linguagem baseada em *XML* (*eXtensible Markup Language*) para descrição de redes.

Uma máquina virtual iniciada pelo *Netkit* é, do ponto de vista de um usuário operador, um computador completo rodando uma distribuição *Debian GNU/Linux*. Para transformar essa máquina virtual em um dispositivo específico basta executar o software adequado (RIMONDINI, 2007).

O *Netkit* é composto por três módulos principais, o *core* (núcleo), o *Kernel*, e o sistema de arquivos (ou *filesystem*). O *Netkit Filesystem* é um arquivo modelo que contém todas as aplicações de uma distribuição Linux padrão, além de diversas ferramentas úteis de rede e servidores de aplicações, além de software para tunelamento GRE (*Generic Routing Encapsulation*) e MPLS (*Multiprotocol Label Switching*), softwares de captura e análise de pacotes como o *tcpdump*, *ssldump*, filtros de pacotes com NAT, ferramentas de segurança, sistema de detecção de intrusão, e vários softwares da camada de aplicação.

O *Netkit* traz também em seu sistema de arquivos padrão o software *Quagga*/*Zebra* que se trata de um software completo de roteamento com os protocolos RIP, OSPF e BGP. O *Zebra* permite que sejam inicializados os módulos de acordo com o protocolo desejados e realizam a atualização automática das tabelas de roteamento, permitindo o estudo do funcionamento destes protocolos. Entretanto, o *Zebra* teve seu desenvolvimento descontinuado em 2005, em favor de um projeto comercial chamado *ZebOS*. Entretanto, um *fork*, denominado *Quagga*, deu sequência ao "extinto" *Zebra* e, nas versões mais recentes do *Netkit* esta versão foi incluída.

Uma alternativa ao *Quagga/Zebra* é o software *XORP* (*Extensible Open Router Platform*), cuja interface de linha de comando foi baseada nas instruções do *JunOS*, sistema operacional da *Juniper* (RIMONDINI, 2007).

O *Netkit Kernel* é um *Kernel Linux* compilado para a arquitetura UML (que será descrita de forma mais completa posteriormente). Junto ao mesmo são compilados vários recursos necessários para os experimentos de rede, porém são removidos recursos que não são úteis, como capacidades multimídia, tornando-o leve e enxuto.

O *Netkit Core* é o conjunto das ferramentas fundamentais do *Netkit*. Enquanto *Kernel* e *FileSystem* são simplesmente elementos da arquitetura UML, e poderiam ser utilizados por quaisquer ferramentas de virtualização baseadas em UML, é este módulo que apresenta seus principais diferenciais. É possível dividir os componentes do núcleo em duas parte: os comandos "V", e os comandos "L".

Os comandos "V" trabalham com uma única instância de máquina virtual, são exemplos os *scripts vstart, vcrash, vhalt*, que permitem a criação, o travamento e o desligamento de uma máquina virtual respectivamente.

Estes arquivos utilizam as ferramentas do UML para instanciar as máquinas virtuais, dimensionar a memória e ativar as interfaces de rede, além de selecionar o console que será utilizado, como *xterm, konsole* ou outra ferramenta.

Os comandos "L", de laboratório, trabalham com uma pasta especial que contém os arquivos *lab.conf, lab.dep, arquivos.startup* e pastas com os nomes das máquinas virtuais. O comando *lstart*, por exemplo, aceita como parâmetro a pasta raiz deste conjunto de pastas, local onde será encontrado tipicamente um arquivo lab.conf. Ao ler o arquivo lab.conf e as pastas, as máquinas virtuais presentes nas pastas/lab.conf serão inicializadas.

Para cada máquina virtual inicializada, um arquivo com o nome da máquina e extensão ".disk" será criado. Este é o **arquivo diferencial**, que contém a diferença entre o arquivo de *filesystem* original e o utilizado pela máquina virtual.

Na pasta do laboratório cada arquivo presente será copiado para o local equivalente do arquivo diferencial, e este será usado em preferência ao original. Por exemplo, considerando o caminho existente no hospedeiro, **`/home/gurgel/labs/lsec`**, como sendo a pasta do laboratório, o arquivo **`/home/Gurgel/labs/lsec/SERVER/etc/apache2/ httpd.conf`** será copiado para **`/etc/apache2/httpd.conf`** do arquivo diferencial da máquina virtual denominada "SERVER" e

será este arquivo que aparecerá na máquina virtual, em detrimento ao existente no *filesystem* padrão.

O arquivo lab.conf é composto de atributos e instruções. Os atributos, iniciados pelo prefixo "LAB_", indicam uma informação, um metadado. São exemplos, LAB_DESCRIPTION e LAB_AUTHOR. Todas as outras instruções aparecerão no formato NOME_MAQUINA[parâmetro]=valor. Por exemplo, dada a máquina SERVER do exemplo anterior, para configurar sua interface eth0 para o domínio de colisão HubA e 256Mb de RAM, são usados, respetivamente, as instruções SERVER[0]=HubA e SERVER[mem]=256.

O arquivo lab.dep contém uma lista formatada que indica a precedência de inicialização das máquinas virtuais. Caso nenhum arquivo exista, as máquinas serão iniciadas sequencialmente conforme encontradas nas pastas. Se o arquivo existir mas estiver em branco, todas as máquinas virtuais serão inicializadas em paralelo.

A indicação de uma máquina, separada por dois pontos e uma lista de outros nomes de máquinas separadas por espaço indica que a primeira precisará aguardar a inicialização de todas as enumeradas para ser inicializada. Por exemplo, dada a seguinte configuração, `CLIENTE1: SERVER FIREWALL`, SERVER e FIREWALL inicializam simultaneamente, e em seguida, é inicializado o CLIENTE1.

Finalmente os "arquivos.startup" são arquivos que recebem o nome da máquina e contém instruções gerais de *script* sendo executados ao final da inicialização da instância virtual.

4.4.2. Outras opções de software

Além do *Netkit*, existem outros softwares que permitem a criação de máquinas virtuais e conexões de rede que não foram utilizados neste trabalho. A primeira das opções utilizada e avaliada é o software *VirtualBox* da *Oracle (2014)*.

O *VirtualBox* é um software de virtualização que permite a inicialização de uma máquina virtual completa para a instalação livre do sistema operacional, provendo a virtualização completa de uma máquina.

O sistema operacional em execução pode possuir interface gráfica, som e todos os demais recursos que teria se estivesse sendo executado nativamente. Ele permite definir uma interface (ou mais) de rede interna e interligar as várias máquinas no mesmo domínio de colisão. Até onde foi possível avaliar, não é possível controlar enlaces individuais com a ferramenta.

Fuertes mostra em seu artigo que o uso do *VirtualBox* para a reprodução de um cenário simples de roteamento é dispendioso pois o processamento consumido e a quantidade de memória RAM ocupada é elevada (FUERTES e VERGARA, 2007), o que torna possível apenas a simulação de pequenas redes.

Outro software de virtualização é o VMWare (VMWARE, 2014), que tem características similares ao *VirtualBox* e cujas diferenças são irrelevantes no escopo deste trabalho.

A última alternativa analisada neste trabalho é o VNUML (FUERTES e VERGARA, 2007). O VNUML foi desenvolvido originalmente no "*Departamento de Ingeniería de Sistemas Telemáticos*" (DIT) da *Universidad Politécnica de Madrid* (UPM) na Espanha. A partir de Janeiro de 2008, o desenvolvimento do *VNUML* foi suportado por uma unidade da Telefônica.

O VNUML tem como característica o uso de uma linguagem baseada em XML para montagem dos laboratórios e o software interpretador do descritor para a execução do cenário. É necessário o acesso de *root* para a configuração de alguns cenários mais avançados. Segundo Fuertes, no cenário de teste o consumo de processador do VNUML foi de 35,14% para o processo de *boot* comparado a 48,16% do *Netkit*. No consumo de memória, o *Netkit* consumiu 185,82 Mbytes enquanto o VNUML, 270,01 Mbytes (FUERTES e VERGARA, 2007).

Porém, entre as funcionalidades que não são oferecidas atualmente por estas ferramentas, tem-se o suporte aos protocolos 3G, TCP móvel, IP Móvel e outros protocolos relacionados com redes móveis. Para a emulação efetiva de uma rede móvel, é necessário que a ferramenta permita a virtualização do hardware adequado.

Os nós móveis de uma rede estão sujeitos ao movimento, o que leva a necessidade de georreferenciar o nó, e sendo assim, por meio do georreferenciamento o nó pode ser localizado no espaço utilizando um mapa de projeção ou sistema de coordenadas. Desta forma, pode ser calculada a distância e a direção de um nó em relação à outra extremidade do enlace (HILL, 2006).

4.5. UML

Esta seção descreve brevemente a arquitetura geral de um sistema operacional Linux e a arquitetura *User Mode Linux* (Linux em modo de usuário), componente principal no desenvolvimento do presente projeto, junto ao *Netkit*.

4.5.1. Arquitetura do sistema operacional Linux

As vantagens do modelo de código livre para o desenvolvimento de sistemas operacionais são amplamente divulgadas pelos colaboradores do modelo que participam ativamente de seus projetos. Entre elas a possibilidade de estudar o código fonte, a constante evolução e o fácil acesso são vantagens atraentes para a comunidade acadêmica.

Dentre os sistemas operacionais de código livre, o Linux ganhou destaque por qualidades como um sistema de arquivos muito eficiente, várias opções de segurança, suporte a redes e bons softwares para servidores, entre outras (FERREIRA, 2003).

Diferentes autores classificam o Linux de diferentes modos: enquanto Tanembaum (2004) descreve o Linux como um sistema operacional de Kernel Monolítico, por ter um Kernel contido em um único arquivo binário e executável, do ponto de vista dos desenvolvedores do sistema, o Linux pode ser considerado como um sistema operacional modularizado (MASTERS e BLUM, 2007).

De fato, ao compilar seu núcleo (*Kernel*) o desenvolvedor do Linux, ou mesmo um usuário com maior experiência, pode optar por modificações de comportamento do sistema operacional. Neste caso, entre as opções é possível até mesmo modificar o algoritmo de escalonamento de processos para um diferenciado. O usuário pode até mesmo optar pela inclusão, ou não, dos *drivers* no núcleo do sistema ou por um módulo que pode ser carregado em uma determinada instância em execução (FERREIRA, 2003).

A possibilidade de carregar e descarregar módulos permite ao Linux a inicialização e o desligamento de dispositivos em tempo de execução. Isto possibilita o desenvolvimento dos próprios módulos, que, uma vez guardadas as precauções necessárias, permitem a troca por módulos de novas versões para correção de falhas ou acesso a novos recursos de um determinado dispositivo. As precauções em questão estão relacionadas com a estabilidade do sistema operacional, uma vez que os módulos são executados em modo privilegiado. Os módulos de dispositivo Linux são equivalentes aos *drivers* de dispositivos em arquitetura Windows.

As funcionalidades e chamadas de sistema do Linux são divididas em dois modos de operação, denominados: modo de usuário e modo privilegiado, também conhecido como modo *Supervisor* ou modo *Kernel*. Essa separação é necessária por segurança, para proteger o funcionamento do sistema operacional de operações indevidas, sejam intencionais ou não, praticadas por usuários conectados ao sistema.

Desta maneira, um programa em modo de usuário que tenha seu funcionamento prejudicado, por exemplo, por um travamento causado por um laço infinito (*loop* infinito), pode ser interrompido pelo próprio usuário ou pelo administrador do sistema sem maior prejuízo aos demais usuários do sistema.

O travamento do processo também não impede o escalonador de trocar os processos em execução nem mesmo o acesso aos dispositivos conectados. Um módulo por sua vez é executado em modo privilegiado e, desta forma, é importante que o desenvolvedor do mesmo seja cauteloso para não provocar maiores danos ao funcionamento do sistema.

Um elemento importante da arquitetura do sistema operacional *Linux* é o acesso a dispositivos por meio de arquivos, geralmente localizados no diretório "/dev", ou subdiretório do mesmo.

Um arquivo de "/dev" representa um dispositivo real ou virtual acoplado ao equipamento que esteja executando a instância do sistema operacional Linux. Desta forma, *syscalls* típicas de uso de sistemas de arquivos, como *open, close, read e write* podem ser utilizadas também para a comunicação com tais dispositivos. Neste caso, tem-se como exemplo uma impressora matricial, que possui um módulo baseado em caracteres.

O desenvolvedor de um módulo deve ser cauteloso para não introduzir vulnerabilidades em um módulo por meio da "derreferenciação"[26] de um ponteiro utilizado no modo de usuário. Sendo assim, chamadas de sistema para transferir dados entre modo supervisor e modo *Kernel* estão disponíveis (CORBET et. al., 2005).

O desenvolvimento de um *driver* em sistema Linux é realizado na linguagem C, com a exceção de uso de funções que fazem parte das denominadas chamadas de sistema no modo de usuário que não devem ser utilizadas quando o software do *driver* operará no modo supervisor, ou modo de *Kernel*. Deste modo, em vez de usar, por exemplo, a função *printf* para saída de texto, a função *printk* será utilizada, sendo esta uma função análoga à *printf* definida no arquivo "*hernel.h*".

Além disso, o sistema operacional *Linux* trabalha com 3 categorias de *drivers*: os primeiros que são os *drivers* de caracteres que operam com serializações de dados; em seguida temos os *drivers* de blocos que operam em dispositivos que possuam um *buffer* e; finalmente, os *drivers* de interfaces de redes que possuem características de bloco no envio e recebimento de

[26] Neologismo, versão aportuguesada do termo "dereference" e do espanhol "desreferencia", ocorre quando uma região de memória é lida diretamente.

pacotes, mas cujas aplicações e protocolos possuem características de transmissão e recepção em fluxo ou *streams*, como os *drivers* de caracteres (CORBET et. al., 2005).

4.5.2. A Arquitetura UML

A virtualização em espaço de usuário obtida por meio do UML se refere na prática a portabilidade do *Kernel Linux* para uma arquitetura específica, denominada, arquitetura UML, que desde 2002 é desenvolvida por seu criador, Jeff Dike, e incorporada no repositório da versão oficial do *Kernel* mantida por Linus Torvalds.

É uma arquitetura no sentido que, todo o código dependente de plataforma, escrito para hardwares específicos como a arquitetura *Intel x86, amd64, spark, arm*, ou outra suportada pelo *Linux*, é substituída por chamadas de sistema para um *Linux* hospedeiro (DIKE, 2005). Desta forma, um *Kernel* UML é obtido por meio da compilação do próprio *Kernel* Linux especificando a arquitetura "um" (*User Mode*) como arquitetura desejada.

Como descrito previamente, a tarefa de desenvolver módulos não é tão simples, uma vez que alterações e testes no *Kernel* podem causar instabilidades na máquina, causando dificuldades de testes e oferecendo barreiras na produtividade. A virtualização por meio da UML permite que um sistema operacional virtual seja instanciado, o que, entre outras vantagens, permite que o *Kernel* em execução e módulos acoplados sejam depurados sem a interrupção da máquina hospedeira.

Cada máquina virtual UML é executada em um espaço de endereçamento de *Kernel* separado, o SKAS (*Separated Kernel Address Space*), que prove proteção e segurança isolando os processos das máquinas virtuais. A principal desvantagem do UML é o desempenho menor, quando comparado com outras ferramentas de virtualização, como o Xen, por não ter suporte ainda a virtualização em nível de hardware provido pela arquitetura x86.

Finalmente, é importante observar que o propósito primário do UML não é ser uma ferramenta de virtualização para uso por provedores de serviços, mas ser uma ferramenta de desenvolvimento dos módulos e do próprio *Kernel* do *Linux*. Um efeito interessante colateral é a possibilidade de montar as redes virtuais por meio do mesmo, o que viabiliza sua utilização como virtualização de redes.

4.6 Trabalhos correlatos

Alguns esforços têm sido feito no sentido de fornecer este tipo de ambiente de experimentação. O projeto EMWIN (*emulating a mobile wireless network using a wired network*), de Ni e Zheng (2002) utiliza uma camada adicional no nível do enlace que permite simular os protocolos de mobilidade em uma rede cabeada. O presente trabalho se diferencia pois não incrementa a camada de enlace com uma sobrecarga que não existiria nos sistemas reais, e tampouco considera mobilidade.

O projeto Orbit, de Raychaudhuri (2005), fornece um serviço que utiliza um *framework* de desenvolvimento de experimentos sem fio, aliado a uma grade quadrada de 64 pontos de acesso sem fio, com distância aproximada de 1 metro entre os pontos (RAYCHAUDHURI et. al., 2005). O trabalho de Singhal et. al. (2006) demonstra uma avaliação do comportamento da virtualização baseada em UML rodando sobre os equipamentos do projeto Orbit (SINGHAL et. al., 2000). Este trabalho utiliza uma infraestrutura real, porém fixa, de redes sem fio, ou seja, também não considera experimentos onde a mobilidade seja importante.

No projeto *Wireless* Gini, Liao (2006) avança um pouco mais, fornecendo uma ferramenta capaz de realizar a virtualização de uma rede Ad Hoc sem fio (LIAO, 2006). No entanto, a página oficial da ferramenta não é atualizada desde 2009. O presente trabalho se diferencia do Gini, pois estendem as capacidades para mobilidade, em vez de prover apenas as conexões às redes 802.11.

4.7. Considerações sobre mobilidade

Como descrito nos trabalhos relacionados, considerando a abordagem de virtualização de redes nenhuma das ferramentas descritas permite que as máquinas virtuais sejam deslocadas em um espaço virtual, de modo que a variação do sinal seja mensurada.

Será possível por meio de tais ferramentas, em particular do NetkitG3, desenvolvido neste projeto, realizar estudos de protocolos de roteamento de redes móveis que levem como métrica a posição, ou a previsão da posição de um nó móvel durante a comunicação; desafios de comunicação entre veículos (cuja proximidade pode ocorrer em uma janela de tempo de poucos segundos); avaliar estratégias de distribuição de cargas de processamento entre nós em movimento, entre outras.

É importante lembrar novamente que a proposta e os trabalhos correlatos consideram apenas sistemas de virtualização de redes. Embora alguns simuladores, como o *OPNET* (OPNET, 2012), possam permitir a simulação de redes móveis e o desenvolvimento de protocolos, se tratará de uma implementação especifica para a pesquisa em questão. Na virtualização de redes, o método desenvolvido já é um protótipo que uma vez aprovado, pode ser transferido para testes em ambiente de hardware real.

4.8. Considerações finais

Este capítulo apresentou os principais conceitos de virtualização de sistemas operacionais, qual a taxonomia utilizada, e como este conceito foi adaptado para a virtualização de redes. Foram descritas também algumas ferramentas de virtualização, e em particular, foi apresentado o *Netkit*, o motivo para sua escolha como ferramenta de referência a ser estendida e um pouco sobre seu funcionamento. Foram apresentados também alguns trabalhos relacionados, que utilizam abordagem semelhante a deste projeto, porém que não apresentam a característica de mobilidade.

No próximo capítulo, descreveremos o NetkitG3, a ferramenta desenvolvida neste projeto, seu funcionamento, seus recursos e como ela atinge os objetivos inicialmente almejados.

Capítulo

5

5. O Netkit G3

5.1. Considerações iniciais

Após a leitura e compreensão dos capítulos anteriores, é possível relacionar os recursos desejáveis em uma boa ferramenta de virtualização de redes de computadores e que não são cobertos pelas ferramentas de virtualização existentes.

É importante relembrar que, embora seja reconhecido que existam diferentes ferramentas de simulação com funcionalidades que viabilizem o desenvolvimento de módulos de comunicação, a diferença principal entre os simuladores e as ferramentas de virtualização de redes é a possibilidade de instalação e utilização dos softwares reais.

Com foco neste requisito, este capítulo apresenta detalhes sobre o desenvolvimento do projeto de mestrado, algumas estratégias de implementação dos recursos de mobilidade e georreferenciamento e as adaptações necessárias na ferramenta para contemplar as funcionalidades previstas.

5.2. Restrições assumidas

Algumas restrições precisaram ser assumidas para a execução do projeto, principalmente para minimizar uma possível complexidade de uso.

Propagação de antenas: O projeto assume que todas as antenas são isotrópicas ideais, isso significa que dada a coordenada de um nó da rede, sua antena é um ponto no espaço R^3 e a propagação do seu sinal possui formato esférico, ou seja, a atenuação do sinal é dada diretamente em função da distância entre as antenas.

$$f(d, f, k) = k(20 \times \log_{10}(d) + 20 \times \log_{10}(f) + 32{,}45) \quad (5)$$

A função de atenuação de propagação em espaço aberto utilizada é dada pela Equação 5, onde d representa a distância linear em metros, e f representa a frequência do canal em

Megahertz (SKLAR, 2005). A potência do sinal é dada simplesmente pela subtração da potência transmitida pelo resultado da equação (5) para os parâmetros informados.

Em relação à fórmula padrão, foi introduzida o multiplicador "K". O valor de K deveria ser obtido então de acordo com a posição dos 2 objetos no espaço, a orientação, a polarização e a forma de propagação das antenas de transmissão, entretanto, como só a antena isotrópica foi implementada, ao invés do valor de K ser calculado, é retornada sempre o valor constante 1 (um).

Não só o cálculo de K pode ser desafiador para antenas não ideais, como permitir que o usuário definisse o espectro de propagação de uma antena arbitrária tornaria a configuração do laboratório penosa e além do uso para o qual foi intencionado.

Restrições de modelagem: Para simplificar o cálculo das coordenadas e linhas de visada, a terra é considerada como uma espera perfeita com raio de 6371 Km, em vez de ser considerado o elipsoide WGS84. Os nós estão sempre em altitudes positivas, em relação ao raio da esfera terrestre.

Restrições de mobilidade: O modelo de mobilidade permite apenas movimento retilíneo uniforme, com a definição de um vetor de velocidade, descrito por meio de uma quadrupla de valores (Vh, Vv, Vz, t), onde Vh é a velocidade linear longitudinal, Vv latitudinal e Vz vertical. As três velocidades são informadas em metros por segundo, enquanto a variável t especifica a quantidade de segundos que o movimento será feito. O usuário pode enfileirar várias entradas de mobilidade para reproduzir um caminho que percorra *waypoints*. Ao terminar o movimento, se não houver outro enfileirado, o objeto parará instantaneamente (Vh, Vv e Vz = 0). Aceleração, desaceleração e curvas podem ser parcialmente simuladas como várias entradas de mobilidade retilíneas, porém, a amostragem de movimento fará o objeto descrever os passos retilíneos em vez de uma curva suavizada.

A velocidade de variação vertical é considerada em relação a altitude relativa, ou seja, caso o objeto percorra grande distância longitudinal, ele acompanhará a superfície terrestre. Vx, Vy e Vz aceitam valores de ponto flutuante, negativos ou positivos. O parâmetro tempo é inteiro, não aceitando frações de segundo.

Tipos de posicionamento: Existem dois tipos de posicionamento, o absoluto e o aferido. O posicionamento absoluto é dado pelos atributos registrados no sistema e são exatos, ou seja, o objeto está na posição virtual (x,y,z) descrita. A mobilidade é realizada por meio da mudança destes atributos. O posicionamento aferido é o que pode ser obtido por meio da leitura do GPS

de uma máquina virtual e possui um erro aleatório, de até 15m de distância na altitude e 5m no posicionamento.

Tratamento de colisões: O sistema não faz tratamento de colisões de posicionamento, permitindo que duas antenas sejam posicionadas na mesma coordenada simultaneamente. Porém, a altitude relativa de cada objeto não pode ser negativa, sendo que a velocidade vertical será zerada caso exista velocidade vertical negativa que ultrapasse a altura de 6731Km da origem do espaço R^3, o que representa o chão virtual.

Neste caso, a função de cálculo de distância retorna como menor distância possível o valor de 0,001 metro, que em uma frequência de 2500MHz produz uma atenuação de 40 dBs.

Ausência de reflexão, refração ou atenuação por obstáculos: O único modo que o usuário dispõe de interferir na atenuação do sinal é por meio de um obstáculo, denominado parede, o qual ele pode definir um valor arbitrário de atenuação em dB. Se o segmento de reta entre transmissor e receptor cruzar esta parede virtual, a atenuação é aplicada, independente do ângulo de incidência, ou seja, sem qualquer refração ou reflexão. O mesmo é aplicado se o transmissor/receptor for colocado no espaço interno do obstáculo. Se o nó móvel estiver em rota de colisão com uma parede, ele irá atravessá-la sem alteração em seu movimento.

Restrição de interferência: A interferência entre roteadores com frequências próximas ou mesmo sobreposição de canais foi modelada com um pequeno aumento no fator de atenuação, sem compromisso no entanto, com a atenuação que seria obtida no ambiente real no mesmo cenário. Isso se deve a dificuldade de obter parâmetros de controle que possam replicar o experimento em ambiente real com os mesmos resultados. Sendo assim, foi desenvolvida *arbitrariamente*, uma função de interferência, descrita na Equação 6:

$$i(rx, p) = \sum_{i=1}^{N} (p \times f(d, f, k)) \qquad (6)$$

A interferência *i* do receptor (*rx*) é dada pela somatória de um percentual *"p"* da intensidade de sinal recebida de todos os roteadores no alcance de *rx* que não seja o roteador em que esteja ocorrendo comunicação. Para ser considerada no alcance, a intensidade do sinal recebido precisa ser maior que a sensibilidade da interface.

O valor de P será de 0,10 (10%) se os canais do transmissor de interferência e do transmissor padrão forem o mesmo. Será de 0,06 (6%) se for um canal adjacente, 0,03 (3%) se a

distância numérica entre os canais for 2, por exemplo, canais 1 e 3 e 0,01 se a distância numérica entre os canais for 3, por exemplo, 1 e 4.

Operadoras 3G: Não houve intenção de viabilizar qualquer tecnologia de comunicação entre diferentes operadoras, sendo assim, o sistema assume que todas as antenas são da mesma operadora.

Sem tun/tap *wi-fi*: Não foi planejado e até o momento não foi percebida justificativa para que uma interface WLAN ou 3G fosse utilizada como ponta de uma conexão *tun/tap*, que gera uma interface compatível com a *Ethernet* para a comunicação entre hospedeiro e máquina virtual. A interface *tun/ta*p ainda pode ser criada tradicionalmente, com a possibilidade, inclusive, de adicionar mais de um tun/tap ao laboratório virtual em execução.

5.3. Desenvolvimento do projeto

A realização de um experimento completo com suporte a georreferenciamento e mobilidade exige a definição de algumas entidades que devem participar da simulação. Os *hosts* fixos e sua configuração operacional, os agentes móveis e seus padrões de movimento, bem como os elementos da infraestrutura.

Serão apresentados brevemente os módulos desenvolvidos e depois um detalhamento mais aprofundado sobre as funcionalidades e funcionamento de cada um deles.

O primeiro módulo desenvolvido, nomeado como **netkitcontrol**, é um módulo de administração das instâncias UML em execução. A versão original do *Netkit* dispensa o uso de um *daemon* de gerenciamento nas máquinas, pois o estado de cada máquina virtual é auto contido e informações temporárias são salvas em arquivos do espaço de usuário. O *netkitcontrol* executa localmente os comandos *vstart*, *lstart* e correlatos do *Netkit*.

Foi necessário desenvolver os módulos adicionais, *uml-gps*, *uml-netwifi* e *uml-net3g*, sendo que cada um dos mesmos requer duas versões, uma como *driver* para a arquitetura uml e um módulo de software para a máquina hospedeira, integrados aos demais serviços.

Um dos módulos do pacote *uml-utilities*, o *uml-switch* foi adaptado para interpretar os modos de operação *wifizone* e *3gzone* além do *hub* e *switch*, permitindo ainda o *handover* entre pontos de acesso para os modos *wifizone* e *3gzone*.

Um módulo *uml-cell* foi criado, com a função de controlar as antenas de uma operadora 3G. O módulo *uml-cell* é responsável pela administração das diversas antenas 3G e pela implementação do CoA, bem como pelo roteamento interno da operadora de telefonia simulada. A comunicação ainda é feita pelo módulo *uml-switch*.

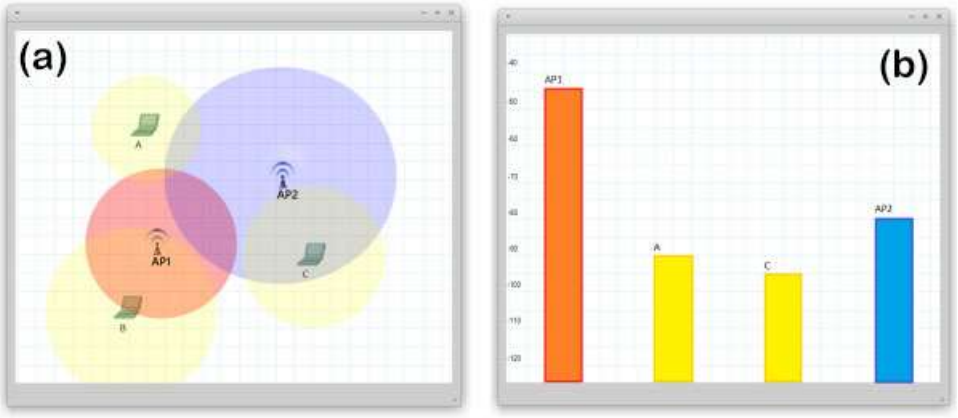
Foi preciso desenvolver um conjunto de ferramentas para ser disponibilizado no *filesystem* Linux que acompanha a ferramenta de virtualização, para obter informações adicionais dos *drivers*, via *mconsole*, que não são obtidos pela interface padrão do *Linux*. Tais ferramentas compreendem *nkgps-get, nkgps-log, nk3g-show*.

Um protocolo de controle precisou ser desenvolvido para especificar todos os comandos de operação e mobilidade do *netkitcontrol*. Além disso, o utilitário de linha de comando *netkitcmd* foi desenvolvido para se comunicar com o *netkitcontrol*.

Duas versões de *Kernel* e *filesystems* foram montadas para utilização. Uma baseada na distribuição Debian versão 7.0 com *Kernel 3.10*, versão do *Kernel* com suporte até outubro de 2015 e outra no software *Openwrt*, utilizado em roteadores sem fio como sistema alternativo ao *firmware* padrão, e baseado em Linux. Os *Access Points* são, basicamente, máquinas virtuais com o *Openwrt* versão 10.03 com adaptação do *driver uml-netwifi*.

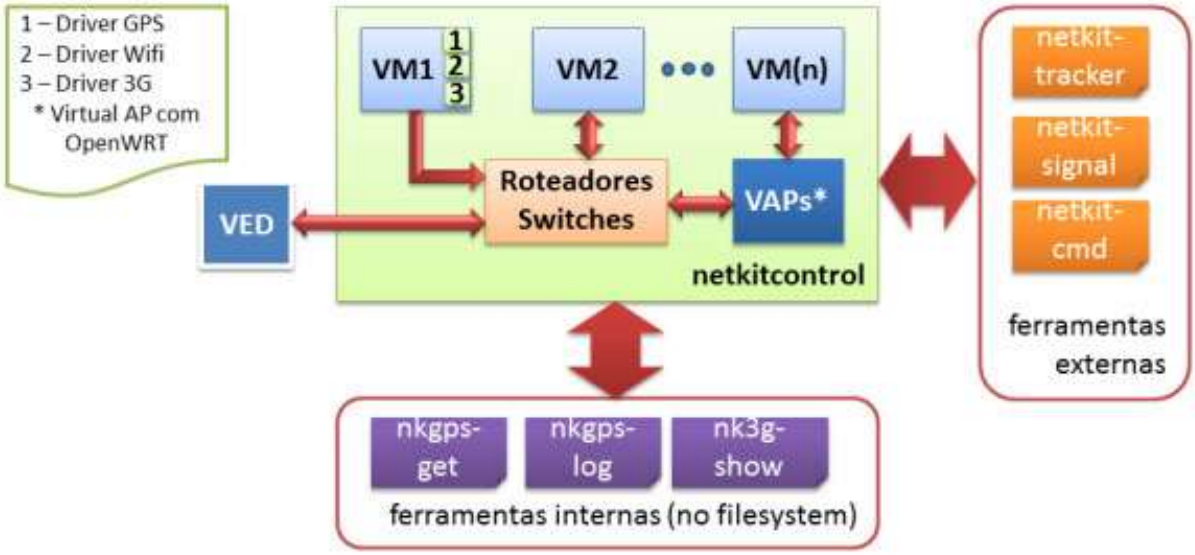
Três pequenos utilitários para uso no hospedeiro foram desenvolvidos. Os utilitários chamados *netkit-tracker* e *netkit-signal* trabalham com a visualização do estado da rede e o *uml-cell-show*, com a consulta de informações sobre o funcionamento da rede 3G. O *netkit-tracker* exibe um mapa dos pontos da rede, exibindo o gráfico de propagação de sinais. O *netkit-signal* mostra duas opções de gráficos de intensidade de sinal em um determinado nó. As imagens geradas por esta ferramenta são ilustradas na **Figura 6**.

***Figura 6** a. Mapa dos nós da rede mostrando a cobertura do sinal dos pontos de acesso e dos transmissores dos nós A, B e C. b. Gráfico de intensidade de sinal sendo recebido pelo nó B.*



Desta forma, na **Figura 7** é ilustrado o diagrama geral dos módulos do projeto.

***Figura 7:** Diagrama dos módulos do NetkitG3*



5.3.1. Funcionalidades do netkitcontrol

O **netkitcontrol** possui as seguintes funcionalidades:

- ✓ Gerencia as máquinas virtuais em execução, mantendo informações sobre seus atributos e enlaces;
- ✓ Realiza a comunicação com as instâncias UML por meio do *driver mconsol*e de uma instância UML;
- ✓ Mantém as posições e o movimento dos satélites virtuais no espaço R^3;

- ✓ Calcula as intensidades de sinal de cada receptor e manipula o *tc* (***traffic control***) adequadamente;
- ✓ Controla a velocidade de recálculo;
- ✓ Fornece um protocolo via *socket* unix e *socket* tcp para extração de informações e controle das máquinas virtuais.

O **netkitcontrol**, desenvolvido em C++, mantêm instâncias de objetos para cada uma das máquinas virtuais, antenas, satélites, células 3G (torres de transmissão). São armazenados a quantidade de memória, o nome, o tipo de nó, os enlaces Ethernet e os enlaces sem fio estabelecidos. O ***netkitcontrol*** é um cliente ***mconsole*** e implementa todas as instruções do protocolo correlato da UML, permitindo a troca de módulos de rede e modificação do enlace, o aumento da quantidade de memória RAM virtual e a interrupção do funcionamento da máquina virtual.

A velocidade de recálculo do estado da rede pode ser definida via protocolo de controle, sendo o padrão de 3 segundos escolhido empiricamente. O estado da rede compreende a posição de cada nó e a potência do sinal recebido pela interface de cada sinal *wi-fi* dentro do alcance, bem como o ajuste da taxa de transferência possível.

O ***netkitcontrol*** interage também com a *uml-switch* e *uml-cell* para as operações de *handover*, reconfigurando os enlaces da rede a cada vez que um sinal de *handover* é recebido dos *drivers uml-netwifi* e *uml-net3g*. Mesmo que um *driver em teste (escrito pelo usuário)*, aplicativo, versão de um aplicativo do usuário em teste não aceite o *handover*, ainda é possível derrubar e reconectar no novo ponto de acesso.

Por meio da interface gráfica ou ferramenta de linha de comando, é possível se conectar ao ***netkitcontrol***, o que permite que a visualização de informações e o acompanhamento sejam feitos em máquina diferente de onde o laboratório esta sendo executado.

5.3.2. Novos drivers

Os novos *drivers* que precisaram ser desenvolvidos precisavam atender as diferentes necessidades do projeto. O *driver uml-netwifi* foi baseado em informações obtidas no código dos *drivers ath9k* (da Atheros) e *rtl-wifi* (Realtek), ambos disponíveis, entre outros, no pacote do

código fonte do *Kernel* do *Linux* [27]. Todavia, a parte específica de hardware foi substituída por operações de comunicação com o módulo *uml-switch* e alguns atributos adicionais, a exemplo do *driver* padrão de rede UML.

O *driver wi-fi* está atualizado para trabalhar com todos os recursos relevantes, como criptografia WEP (*Wired Equivalent Privacy*) e WPA (*Wi-Fi Protected Access*) e criação de redes *Ad hoc*. Para isso, o usuário final pode utilizar as instruções "*iw*" do *Linux*, ou por meio de interface gráfica, caso o servidor X venha a ser instalado no sistema de arquivos virtualizado do usuário final.

O *driver* GPS e o *driver* 3G possuem comportamentos um pouco diferenciados dos *drivers* tradicionais. Um *driver* GPS padrão não faz parte da árvore do código fonte do *Kernel Linux*, sendo assim não há um padrão de comunicação entre diferentes dispositivos. Alguns dispositivos montam como unidades de memória auxiliar para que arquivos com os *logs* de leituras do GPS possam ser coletados e interpretados por diversas aplicações, como o Garmin eTrek. Outros possuem interfaces proprietárias de comunicação que só funcionam com seus softwares, ou com softwares de terceiros que tenham obtido conhecimento do protocolo.

Nos dois casos, o aumento da latência e a perda de pacotes e obtida por meio do pacote tc (traffic control) do Linux. Este pacote, quando a opção de "emulação de redes" é compilada junto ao Kernel, permite utilizar instruções como as seguintes:

1. # **`tc qdisc change dev eth0 root netem delay 50ms 20ms distribution normal`**.

2. # **`tc qdisc change dev eth0 root netem loss 0.2% 40%`**.

A instrução 1 inclui uma latência de 50 milissegundos na comunicação, com uma variação aleatória normal de 20 milissegundos para mais ou para menos. A instrução 2 provoca uma probabilidade de 0,2% de um pacote ser perdido, sendo que o segundo parâmetro de 10% indica que, dado um pacote perdido, a probabilidade do pacote posterior ser perdido também passará para 40%, podendo ser utilizado para que os pacotes sejam perdidos em rajadas.

A opção escolhida foi desenvolver um *driver* serial simples, capaz de retornar a coordenada de posicionamento aferida, ou então gerar um arquivo de *log* com um conjunto de leituras, em formato similar ao formato utilizado pelo dispositivo Garmin eTrek.

Para obter estas informações, o usuário pode utilizar na máquina virtual os aplicativos de cliente *nkgps-get* e *nkgps-log*, que acessam o *driver* do GPS para obter a informação. A

[27] O site oficial do Kernel Linux mantido por Linus Torvalds e algumas outras poucas pessoas é www.kernel.org – acessado em 10/09/2014.

ferramenta *nkgps-get* mostra as coordenadas GPS no formato latitude e longitude, enquanto a ferramenta *nkgps-log* exibe o *log* das últimas coordenadas obtidas pelo GPS virtual.

As coordenadas GPS de cada máquina são calculadas sem utilizar o sistema de equações, acrescentando um erro aleatório à posição relativa da máquina. Porém, o *driver* GPS recebe as informações de distância e posição dos demais satélites, informações que podem ser obtidas posteriormente com a evolução das ferramentas *nkgps-log* e *nkgps-get*.

No caso dos *drivers* 3G, o problema encontrado foi outro. A ausência da outra ponta, das células em uma torre de transmissão e de todo o arcabouço de software utilizado pelas operadoras, entre o *firmware* das antenas e gerenciamento das conexões. Do lado 3G, o *driver* 3G nada mais é que um *driver* USB padrão ligado a uma porta e as aplicações se comunicam com o mesmo serialmente, ocultando do usuário e mesmo do sistema operacional boa parte dos bastidores da comunicação 3G.

Para expor um pouco mais as informações sobre o funcionamento de uma rede 3G, o *driver* foi implementado não como um *driver Ethernet*, mas como um *driver* de *modem* para telefonia seguindo o antigo padrão V.90, que precisa utilizar a configuração de PPP (*Point-to-Point Protocol*) para realizar a conexão. O serviço PPP deve ser então utilizado e uma conexão ppp0 criada.

O utilitário *nk3g-show* mostra informações complementares sobre a célula o qual a conexão está ativa, qualidade do sinal sem fio e outras informações que não são extraídas nativamente pelo PPP.

5.3.3. O uml-cell

O **uml-cell** é uma ferramenta complementar que apoia a ferramenta *uml-switch*, cuidando especificamente do roteamento na rede 3G. Pontos de acesso 3G podem ser adicionados no mapa, com diferentes endereços de IP dentro de uma mesma faixa de rede. O *uml-cell* foi desenvolvido assumindo que existe uma ligação entre todos os pontos de acesso, que é transparente ao usuário. Uma das células, chamada de célula mestre, contém as informações sobre o DHCP do ponto móvel e um segundo IP que pode ser utilizado para conectar a infraestrutura 3G a um roteador de *backbone* do experimento montado.

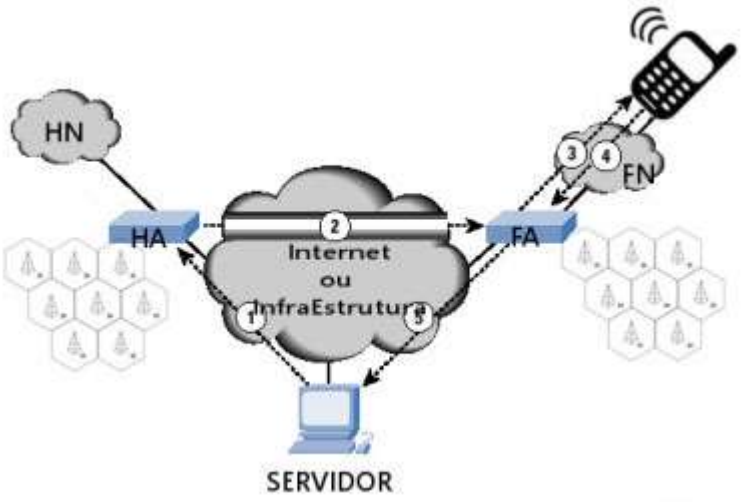
Sendo assim, um nó móvel pode se conectar a qualquer ponto de acesso da infraestrutura 3G. Estes nós não são representados por máquinas virtuais, sendo que o *uml-cell* apenas

configura rotas entre os pontos de acesso. Na **Figura 8** é ilustrado o funcionamento da estrutura 3G comum.

O servidor envia uma mensagem para o nó móvel, o agente interno (*Home Agent* - HA) possuirá o registro do endereço CoA (*Care of Address*) do nó móvel e pela infraestrutura da operadora, ou pela Internet encaminhará o pacote para este endereço, de responsabilidade do agente externo (*Foreign Agent* - FA).

Neste caso, os pontos de acesso 3G não são máquinas virtuais como os pontos de acesso do padrão 802.11, mas encapsulam todo o roteamento descrito na **Figura 8**, viabilizando apenas o *handover* entre as antenas de acordo com as diferenças de sinal virtualmente aferidas, e a comunicação entre os pontos conectados à rede 3G.

Figura 8: *Funcionamento de uma rede 3G*



Fonte: *adaptado de http://www.cisco.com/web/about/ac123/ac147/images/ipj/ipj_4-2/figure1.gif*

A informação de roteamento de um nó, seu CoA e endereços de agentes podem ser obtidos por meio da ferramenta *nk3g-show*, em uma máquina virtual conectada a rede, ou por meio da aplicação de hospedeiro *uml-cell-show.*

5.3.4. O Protocolo de Controle Netkit (NCP)

O Protocolo de Controle *Netkit* (NCP - *Netkit Control Protocol*), também proposto neste projeto, é a ferramenta pelo qual o usuário pode interagir com o **netkitcontrol**. O NCP pode ser utilizado diretamente, por meio de um software cliente escrito pelo usuário final, ou então pela ferramenta *netkitcmd*.

Basicamente, o protocolo inclui as seguintes funções[28]:

- ✓ **Adiciona_nó (nome, memória, tipo):** cria uma nova máquina na memória RAM com 1 console, com o tamanho definido. O tipo pode ser comum ou ponto de acesso *wi-fi* (*Linux* ou *openwrt*);
- ✓ **Desliga_nó(nome):** remove uma máquina identificada pelo nome da memória, desligando-a;
- ✓ **Trava_nó(nome),** faz o mesmo da operação *desliga_nó*, porém via travamento. O arquivo diferencial e todos os arquivos temporários serão excluídos;
- ✓ **Ajusta_memoria(nome, memória):** permite modificar a quantidade de memória de uma máquina UML, sempre para mais;
- ✓ **Instala_rede(nome_origem,nome_destino,nome_dominio_colisao):** cria um enlace com o nome de "nome_dominio_colisão" entre as máquinas virtuais de origem e destino;
- ✓ **Remove_rede(nome_origem,nome_destino,nome_domínio_colisão):** faz a operação reversa do instala_rede, removendo o enlace das máquinas;
- ✓ **Adiciona_console(nome**): adiciona uma nova tela de console (*shell*) na máquina virtual, caso o usuário tenha fechado o terminal que esteja utilizando;
- ✓ **Remove_console(nome, ind):** descarta um *shell* de console da máquina virtual;
- ✓ **Lista_maquinas:** retorna uma lista das máquinas e seus principais atributos;
- ✓ **Movimenta(latitude, longitude, altitude, velocidade):** permite definir o padrão de movimento de uma máquina virtual. Sua execução é imediata e remove todos os elementos da fila de execução;
- ✓ **Empilha_movimento(latitude, longitude, altitude, velocidade)**: permite acrescentar vários elementos de movimento ao ***netkitcontrol***. O primeiro elemento adicionado tem início imediato;
- ✓ **Ajusta_velocidade(tempo):** Ajusta o intervalo de tempo entre recálculos;
- ✓ **Carrega_laboratorio(caminho):** permite carregar uma pasta que contenha o arquivo lab.conf e as pastas das máquinas virtuais;

[28] Na prática, todas as instruções do protocolo foram implementadas em inglês, para que a integração posterior seja facilitada.

- ✓ **Inicia_laboratorio:** inicia a execução do laboratório carregado pela instrução anterior;
- ✓ **Finaliza_laboratorio:** desliga todas as máquinas virtuais do laboratório em execução;
- ✓ **Adiciona_3G(nome_nó, coordenadas, ip):** Acrescenta um ponto de acesso 3G nas coordenadas especificadas e com o IP especificado;
- ✓ **Ajusta_3G(nome_nó, atributo, valor):** permite ajustar o valor de um atributo relacionado ao nó 3G;
- ✓ **Remove_3G(nome_nó):** remove o transmissor 3G identificado pelo nome informado;
- ✓ **Adiciona_VED(nome, pid):** acrescenta um VED (Virtual Embedded Device, descrito com maiores detalhes na próxima sessão) na estrutura na infraestrutura de redes;
- ✓ **Conecta_VED(nome, ip, domínio_colisão):** acrescenta um VED na estrutura na infraestrutura de redes;
- ✓ **Remove_VED(nome):** remove um VED da infraestrutura de comunicação;
- ✓ **Posiciona_elemento(nome, latitude, longitude, altitude):** posiciona qualquer elemento nas coordenadas especificadas;
- ✓ **Le_interface(nome_nó, interface):** permite recuperar as informações de uma interface de rede;
- ✓ **Le_posicao(nome_nó):** permite recuperar a localização de um nó;
- ✓ **Le_satelite(nro):** retorna as coordenadas do satélite solicitado. Ao todo são 20 satélites virtuais.
- ✓ **Adiciona_parede(nome, x1, y1, x2, y2, x3, y3, x4, y4, atenuacao):** Permite adicionar uma parede nas coordenadas descritas, com o fator de atenuação informado.
- ✓ **Remove_parede(nome):** Remove a parede informada.

O *netkitcmd* é apenas um interpretador de linhas de comandos, os comandos do protocolo são passados via parâmetros de linha de comandos, e as instruções são enviadas no formado adequado. Um exemplo de comando é:

#: netkitcmd Add_node --name=nome_maquina --mem=256 --tipo=linux

Todos os comandos do *netkitcmd* obedecem a notação de incluir o nome do atributo com duplo traço e o valor com o símbolo de igualdade em seguida. A resposta

5.3.5. O módulo VED

A estrutura de um VED (*Virtual Embedded Device, Dispositivo Embarcado Virtual*) é ilustrada em maiores detalhes na **Figura 9**.

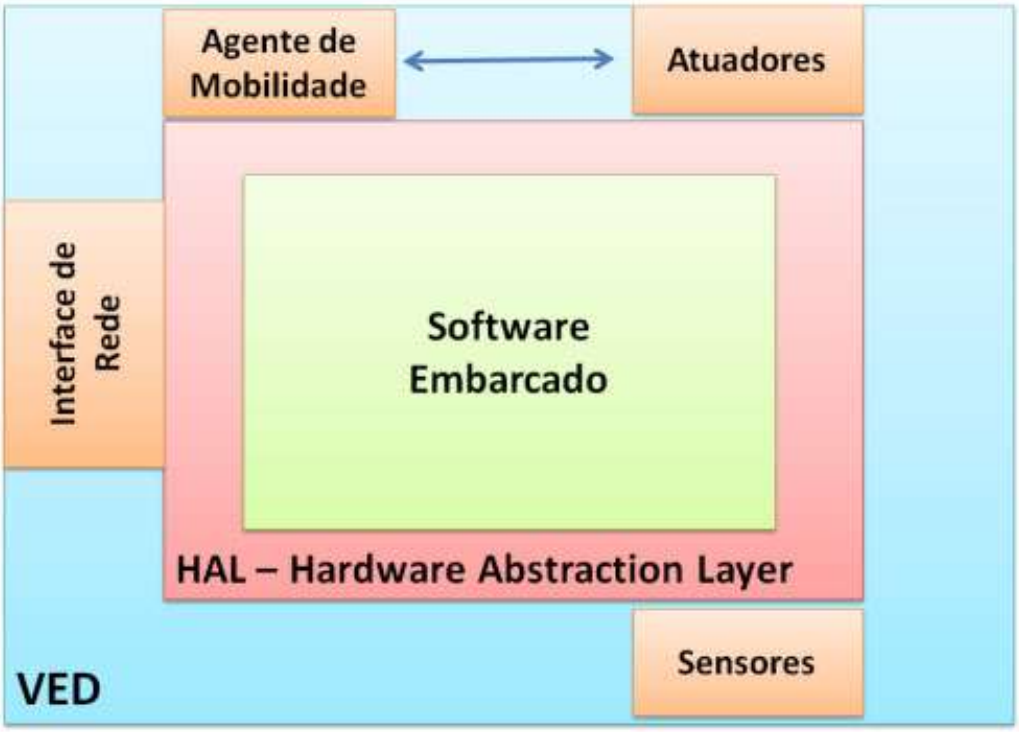
O VED possui agentes programáveis que recebem as informações do HAL (*Hardware Abstraction Layer*) e realizam a atuação com o ambiente, ou que captam informações do ambiente.

Sensores de pressão, GPS, giroscópios, sensores de luminosidade podem ser criados e suas leituras dependem da programação que pode seguir um padrão aleatório ou ser baseada na posição e no tempo.

O software embarcado que precisa ser estudado será então compilado e carregado junto ao HAL. No escopo deste projeto, apenas foi desenvolvido um módulo VED como uma prova de conceito, ficando o desenvolvimento de outros VEDs a cargo da comunidade que tenha interesse na sua utilização.

Foi desenvolvido, no entanto, o arcabouço de softwares que farão a interface do núcleo interpretador do dispositivo embarcado e a estrutura de experimentação. Este arcabouço é denominado **Framework NetkitG3 VED,** ou simplesmente **Framework VED**.

Figura 9: *Estrutura de um VED*



O VED desenvolvido é um sensor de temperatura, que retorna um valor entre 25 e 35 graus Celsius aleatoriamente. A agente de mobilidade interage com o **netkitcontrol** para modificar a posição do sensor, sendo que no caso de testes foi adotado um padrão retangular de movimento. A HAL nada mais é do que uma camada de comunicação que deve ser inserida no fonte do VED que faz a comunicação com os módulos **netkitcontrol** e *uml-switch*.

Sendo assim, um pequeno *daemon* que inclua os *headers netkitved.hpp, netkitvednet.hpp* e *netkitvedmov.hpp,* arquivos do framework VED, podem: utilizar as funções de *netkitvedmov.hpp* para alterar sua posição no espaço R^3 de acordo com a programação desejada, no caso de um dispositivo móvel; utilizar as funções do *netkitvednet.hpp* para realizar a comunicação. O header *netkitved.hpp* já define uma classe que encapsula a criação de um socket, a conexão, e a mobilidade.

É importante frisar que o software embarcado é específico para prototipar os experimentos, mas uma camada de abstração de hardware (HAL) mais avançada, que emule uma arquitetura completa, pode ser desenvolvida para utilizar um software que possa ser realmente transportado para um determinado dispositivo.

Um cenário hipotético para exemplificar o potencial do **Framework VED** seria criar um dispositivo virtual que representasse um robô. Este VED receberia uma pasta de imagens,

tiradas por uma câmera digital simples ou por um robô real. A HAL desenvolvida deveria mapear as instruções de movimentação do código normal em movimentos do Framework VED.

O robô poderia tomar a decisão de movimento baseado nas imagens sendo processadas. O HAL seria o responsável por selecionar as imagens de acordo com as coordenadas. Múltiplas instâncias do VED Robô poderiam ser executadas ao mesmo tempo, cada uma com um acesso a uma pasta de imagens e a comunicação entre os robôs poderia ser testada. É neste teste de comunicação que reside o potencial do **Framework NetkitG3 VED**. O algoritmo de comunicação poderia ser testado, e os movimentos dos robôs avaliados, antes mesmo dos robôs serem colocados em campo.

Destaca-se, no entanto, que não foi feito nenhum teste com um cenário desta complexidade, mas os sensores de temperatura e os testes de movimentar o sensor apresentaram o funcionamento esperado.

5.3.6. Integração com o Netkit

Para completar a implementação, foi necessário realizar algumas modificações no *Netkit*. Embora tenha sido utilizada uma nova versão de *Kernel* e *FileSystem* padrão, os *scripts* de inicialização originais do *Netkit* e toda a preparação do *Filesystem* foram incluídos no ***NetkitG3***.

Os comandos "V" do *Netkit*, como vstart, vlist, foram modificados para interagir com a interface de comandos *netkit_cmd*, para aproveitar os parâmetros adicionais incluídos. Já o próprio *netkitcontrol* utiliza os comandos "L", como *lstart*, para a interpretação do arquivo de configuração e inicialização das máquinas virtuais.

O arquivo de configuração lab.conf foi modificado da seguinte forma. Considerar um nome de máquina virtual chamado MAQUINA:

1. As interfaces de rede, antes descritas como **`MAQUINA[nro_interface]=valor`**, passam a ser identificadas como **`MAQUINA[type, nro_interface]=valor`**. Tipo assume os valores *lan*, *wi-fi* e *3G*. O espaço após a vírgula entre type e nro_interface é opcional, mas não pode ser maior que 1 (restrição da implementação).

2. Foram acrescentados aos atributos especiais "LAB_", como LAB_DESCRIPTION e LAB_AUTHOR, os atributos LAB_XORIGIN, LAB_YORIGIN, que permitem definir latitude e longitude do ponto de origem. Se não especificadas, são consideradas as coordenadas 0, 0.

3. Foi acrescentado o atributo especial "LAB_USEMETER=true", que indica que os valores de posição do arquivo das máquinas virtuais podem ser descritos como quantidade de metros da origem, em vez de coordenadas.

4. O tipo de máquina pode ser especificado com a instrução **`MAQUINA[TYPE]=valor`**, onde TYPE é constante, como em **`MAQUINA[mem]`** e valor pode ser *node, ap, 3gc* ou *ved*.

5. O parâmetro X, Y e Z permitem definir as coordenadas latitude e longitude e altitude em metros da máquina virtual, como em: **`MAQUINA[X]=32.2123`**. Caso seja ativada a *flag* "usemeter", X e Y são descritos como distância em metros da origem, facilitando a criação do arquivo lab.conf.

6. É possível criar um arquivo lab.blocks, descrevendo os polígonos obstáculos. Cada linha deve incluir um obstáculo ou parede, descrito pelo nome da parede, por 4 coordenadas e o décimo parâmetro sendo o fator de atenuação. O exemplo apresenta a formatação recomendada, mas os espaços são opcionais:

   - **`block(muro1, 0,0, 0,10, 10,10, 10,0 ,20)`**

7. Além dos arquivos *maquina.startup*, o usuário pode acrescentar também o arquivo maquina.moves, com as instruções *delay_to_start*, que especifica um tempo, em segundos, antes de iniciar os movimentos, *loop=true* que ativa a repetição dos movimentos, de modo que o objeto voltará para a posição de origem e repetirá a lista de movimentos, e a instrução move, que equivale no protocolo à instrução NCP *empilha_movimento*. Um pequeno exemplo se *maquina.moves*:

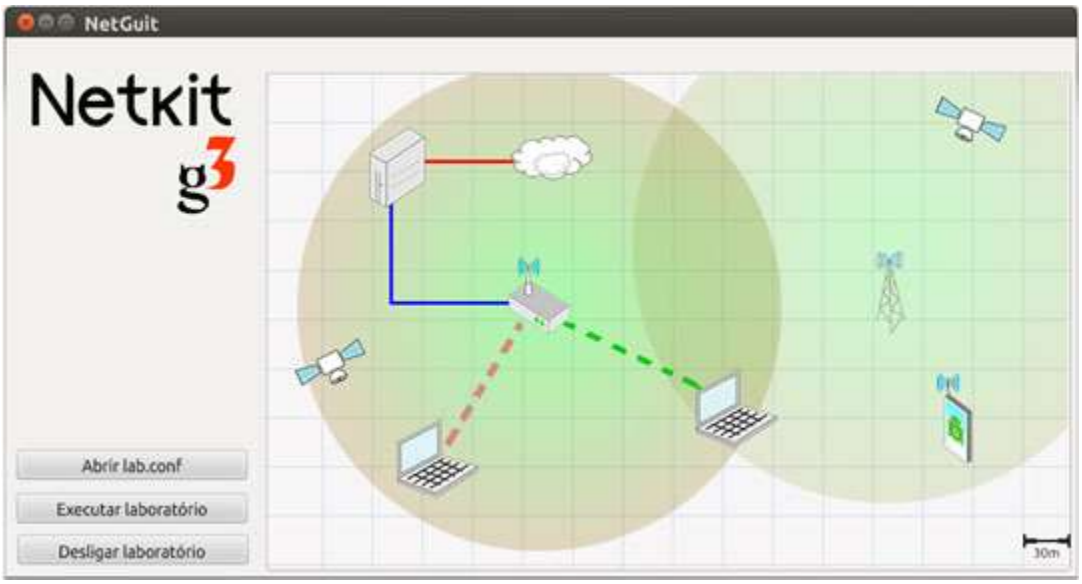
   - **`delay_to_start=0`**
   - **`loop=true`**
   - **`move(0, 0, 0, 0)`**
   - **`move(1, 1, 0, 1)`**
   - **`move(1, -1, 0, 1)`**

5.4. NetGuit

NetGuit (com o u pronunciado, como se houvesse trema) é o nome da interface gráfica de controle que está sendo desenvolvida pelo aluno de iniciação cientifica Guilherme Carvalho, do IFSP (Instituto Federal de São Paulo). O projeto original não previa a construção de laboratórios por meio da NetGuit, porém, com os recursos desenvolvidos, acrescentar tais funcionalidades

começou a parecer natural e rapidamente factível. O nome vem de GUI (*Graphical User Interface*) inserido no meio da palavra *Netkit*, substituindo os caracteres "ki". Quando concluída, esta interface vai substituir e integrar algumas das ferramentas de visualização e controle desenvolvidas.

Figura 10: *Protótipo do NetGuit em execução*



Na **Figura 10**, é possível observar uma imagem de um protótipo do **NetGuit** em execução. São visíveis satélite, antena e torre de celular. A esfera em gradiente verde vermelho mostra o sinal e a região de cobertura. Os enlaces inteiros são conexões cabeadas, enquanto os pontilhados são enlaces sem fio. Faltam na tela o relógio, que mostra o tempo de simulação, os nomes das máquinas e mais algumas opções de visualização, além de controles adicionais.

Esta interface está sendo desenvolvida em Python, utilizando as bibliotecas pyGame para o canvas, e pySide para os demais controles. Para o fechamento desta monografia, foi utilizada uma versão parcial da interface, ligada aos demais módulos desenvolvidos neste mestrado.

5.5. Verificação e validação

5.5.1. Experimentação com equipamentos reais

Para efetuar os testes, cenários de experimentação foram desenvolvidos, como a criação de redes *ad hoc*, redes com múltiplos roteadores em WDS (*Wireless Distribution System*) e o *handover* de um *notebook* passando por eles.

A validação foi feita comparando os resultados do simulador com a replicação destes experimentos em equipamentos reais. Foram utilizados os seguintes equipamentos:

- 1 Notebook Core i5 3230M, 6Gb de RAM, Mageia 4 Custom, Rede Wifi Intel 802.11n
- 1 Notebook Core i5 3210M, 4Gb de RAM, Mageia 4 Custom, Rede Wifi Intel 802.11n
- 1 Notebook Core i5 3317U, 6Gb de RAM, Windows 8, Rede Wifi Intel 802.11n
- 5 Roteadores TP-Link WR941NDv5.0 com OpenWRT Attitude Adjustment 12.09 final
- 2 Celulares Motorola G XT1033, Android KitKat 4.4.3/4.4.4
- Software Android Wifi Analyser 3.7.x / 3.8.x
- GPS Garmin eTrek
- Modem 3G Claro Huawei

Sobre o Mageia Linux, todos os pacotes foram atualizados, a versão do Kernel utilizada, todavia, foi a 3.13.9, versão congelada também para o NetkitG3 utilizado nos experimentos.

De modo geral, a quantidade de problemas observados na experimentação virtual é menor que a quantidade de problemas experimentados com equipamentos reais, fato explicado devido a inexistência das fontes de interferência alheias ao cenário estudado na versão simulada.

Em cenários de alta densidade de equipamentos, onde não há controle sobre artefatos que possam interferir na comunicação, diferentes replicações dos experimentos apresentam resultados diferentes. Serão detalhados os quatro principais experimentos realizados:

Experimento 1 – Rede *ad hoc* em pequeno espaço;

Experimento 2 – Mobilidade e redes *ad hoc*;

Experimento 3 – Repetidores WDS;

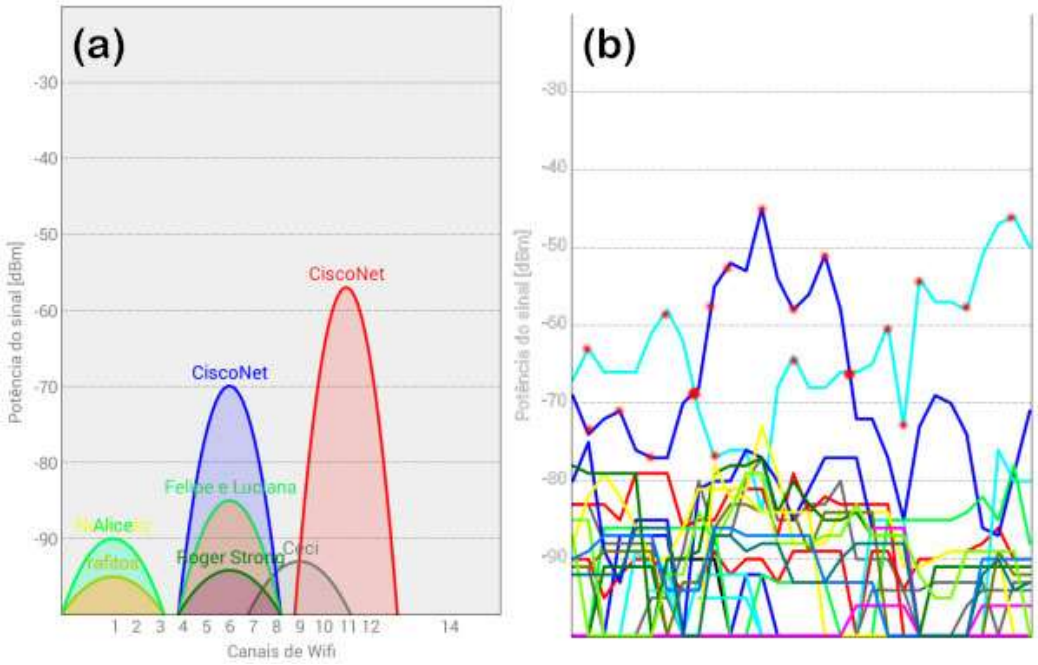
Experimento 4 – Redes com maior densidade.

Estes experimentos foram replicados no **NetkitG3**.

Uma consideração importante é sobre as métricas obtidas. O desvio padrão da intensidade de sinal é muito variável. A observação por alguns minutos de um sinal *wi-fi* em particular, permite observar uma variação na ordem de 3dB, sem que nada diferente aparentemente ocorra.

A mudança do celular em relação ao ponto de acesso em alguns centímetros é suficiente para levar essa variação para até 10dBs. Uma variação de 6Db positiva indicaria um ganho do dobro da intensidade, enquanto uma variação negativa, um corte pela metade. Na **Figura 11** é ilustrada essa variação colhida em um intervalo de 2 minutos.

Figura 11: *Processo de handover, em 10.a, a intensidade do sinal é destacada na ferramenta Wi-Fi Analyzer, em 10.b, foram destacadas com pequenos nós as linhas de intensidade de sinal* [29].



Neste caso, o sinal em destaque, muito acima dos demais, é um sinal de um roteador a aproximadamente 2 metros de distância do dispositivo, e a variação de 10dBs foi provocada pela troca do celular, um Moto G, de posição relativa de altitude, mantendo a distância aproximada. Pela teoria, o dobro da distância implicaria em uma variação de 6dB. Os outros sinais da imagem são pontos de acesso do local o qual não se tem controle, uma vez que os experimento não foi realizado em um local sem interferências ou com interferências controladas.

Geralmente, em cenários próximos, onde a relação sinal/ruído é mais alta e a intensidade do sinal é bastante superior ao limite de sensibilidade do equipamento, as taxas máximas de transmissão são obtidas com pouca variação.

Em cenários onde os dispositivos são posicionados em posições cujas fontes de ruído, bloqueios, sejam suficientes para provocar uma variação mais intensa da relação sinal/ruído cuja atenuação pode ultrapassar os limites inferiores de sensibilidade, a taxa de transmissão também sofre forte variação.

[29] Estas figuras tiveram suas cores invertidas a partir da imagem capturada pelo sistema Android, cujo fundo original é escuro.

Em todos os casos, a intensidade do sinal foi monitorada pela ferramenta *Wi-fi Analyzer*, disponível gratuitamente para celulares *Android* que permite armazenar as variações de intensidade de sinal.

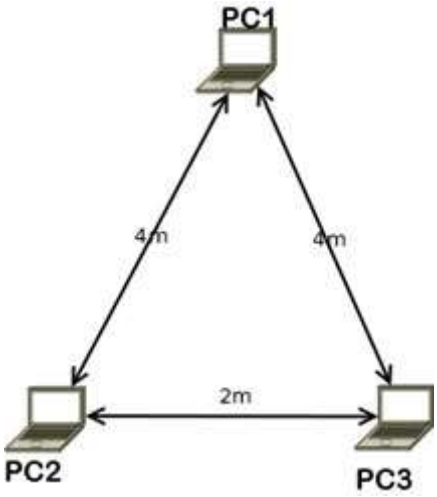
5.5.2. Rede ad hoc em pequeno espaço

Este experimento foi realizado com três notebooks em duas residências diferentes, de acordo com as especificações fornecidas anteriormente. Uma rede *ad hoc* com 3 pontos foi criada, todos em visada, para verificar a funcionalidade, modo de configuração e foi aferida a taxa de transferência obtida. Um arquivo de vídeo de 200 Mb foi transferido 10 vezes, com pequenas variações de posição dos computadores. A distância entre os pontos está descrita no diagrama.

As taxas de transferência obtidas foram na faixa de 10Mb por segundo, sendo a transmissão sempre efetuada 2 a 2, sem concorrência pelo canal. O objetivo deste experimento foi comparar a configuração dos notebooks com a das máquinas virtuais.

No caso da transferência nas máquinas virtuais, a taxa de transmissão obtida foi mais baixa, em média 2 Mb/s. Isso é explicado pela concorrência que ocorre na virtualização pelo dispositivo de armazenamento, sendo as operações de entrada e saída um gargalo. A **Figura 12** ilustra o diagrama do cenário.

Figura 12: *Cenário de rede Ad-Hoc*



5.5.3. Mobilidade e rede ad hoc

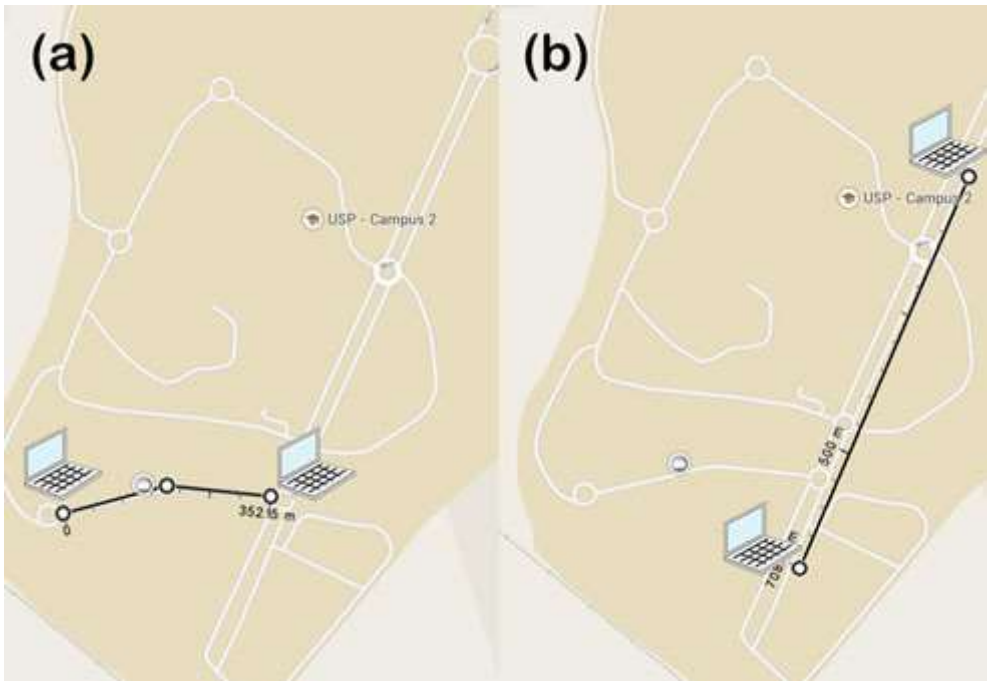
Este experimento foi realizado no Campus II da USP de São Carlos, com a participação e colaboração dos membros do laboratório de pesquisa. O objetivo foi avaliar experimentalmente os efeitos das velocidades na faixa veicular (40 a 120 Km/h) na comunicação *Wireless.* Para este experimento, foram utilizados dois notebooks Intel Core 2 duo, 4 Gb de RAM e placas de rede sem fio Intel, compatíveis com rede 802.11n e Windows 7. As velocidades são relativas, sendo utilizados dois carros que atingiram no máximo 60 Km/h, se aproximando e se afastando pelas vias opostas, atingindo então uma velocidade relativa de 120Km/h máximo.

Cada uma das seções de teste apresentou uma pequena diferença, como o local exato (feito em 4 posições do Campus), e fatores derivados dessa mudança, como a distância de proximidade atingida. A ordem de velocidade utilizada não permitiu obtenção direta da variação da potência de sinal, pois as distâncias envolvidas eram pequenas e o tempo de execução curto.

Sempre que foi respeitada a visada entre os computadores, transportados pelo passageiro, a comunicação foi feita com sucesso. Sendo os transmissores dos notebooks de potência baixa, poucos bloqueios foram suficientes para interromper a comunicação. A amostragem neste experimento foi através do programa "ping", registrando os pontos onde ocorria a perda de visada. Este experimento foi replicado apenas para obtenção do comportamento, entretanto os dados coletados foram aproveitados neste trabalho.

Este cenário foi reproduzido no NetkitG3, as máquinas virtuais colocadas em movimento, e os pings efetuados, utilizando movimentos na faixa de 12m/s (43.2Km/h) e 34m/s (122.4 Km/h). Este cenário foi reproduzido com o uso de paredes para simular um bloqueio de visada. Sem bloqueios, não houve perda de sinal, ou aumento sensível da latência, que só foi introduzida com o uso de bloqueios. A Figura 13 ilustra os percursos efetuados (as distâncias são aproximadas).

Figura 13: *Percursos - Campus II*

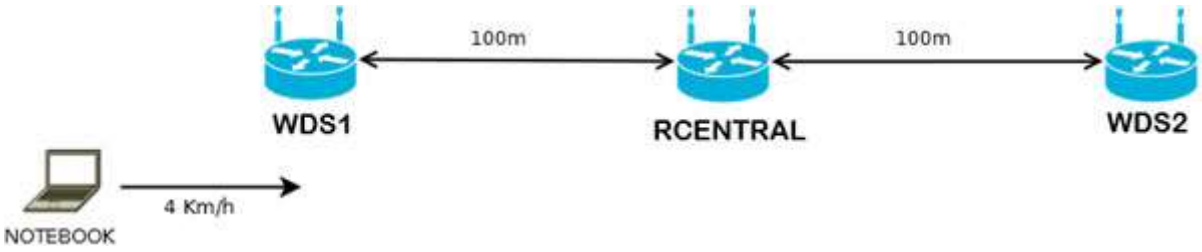


5.5.4. Repetidores WDS

Neste cenário, foram utilizados 3 roteadores e a configuração feita para que existisse um roteador central e 2 repetidores, colocados a 100 metros de distância cada (200 lineares) e com potência de transmissão 3 dB. Em algumas das repetições, o teste mostrou perda de pacotes ICMP (*Internet Control Message Protocol*) durante o *handover*, mas isso pode ter sido causado pela baixa potência do sinal. No caso de potências maiores, o sinal do roteador mais distante ficava acima do limite de sensibilidade do nó, sendo que o *handover* não era necessário e difícil de monitorar.

A perda de pacotes e a latência apresentada neste cenário foram um pouco maiores do que os obtidos na prática, porém o comportamento geral foi similar, satisfazendo os objetivos iniciais. A **Figura 14** ilustra o cenário deste experimento.

Figura 14: *Cenário de repetidores WDS*



5.5.5. Rede com maior densidade

Esta rede foi montada em espaço semiaberto. Este experimento foi realizado em ambiente não controlado, e contou com 5 roteadores, sendo colocados próximos às janelas de diferentes blocos e andares do condomínio de residência do autor. Os 5 roteadores formavam 2 redes, a *CiscoNET*[30] e a *CiscoTESTE*, ambas com criptografia WPA2 para evitar a conexão por terceiros.

Os roteadores de uma mesma rede ficaram em faces opostas dos prédios, ligados por cabos, sendo que em uma das redes um dos roteadores, com seu *wi-fi* desligado, foi utilizado como central. Além disso, o condomínio apresentava inúmeros outros roteadores em diversos canais, sendo as leituras demonstradas na **Figura 11** apresentada previamente.

Nos dois casos, foi ligado um notebook com IP configurado, sendo a CiscoNET configurada com a série 192.168.1.0/24 e a rede *CiscoTESTE* com a série 10.1.1.0/24, com os notebooks ativos com *Linux*, Servidor *DHCP* Apache e um arquivo de vídeo com 200Mb. Foram escolhidos 6 pontos e 4 percursos em linha reta, com pontos de partida iniciais e finais bem definidos. Foram feitas 3 replicações de cada um destes, totalizando 30 operações de downloads. Nenhum dos percursos permitia a manutenção da visada entre os dois roteadores, tentando neste caso minimizar a ausência de contato. Após reestabelecida a conexão com o segundo roteador, era necessário reiniciar o wget com a opção "-c" que permite continuar o download interrompido.

Para reproduzir este comportamento no **NetkitG3**, foi necessário adicionar as paredes virtuais, sendo este experimento o motivador da criação das mesmas. Exceto pelas taxas de transmissão menores, o comportamento obtido foi similar, exceto pela inexistência de ruído detectado em alguns pontos.

[30] Cisco é o nome do gato da família do autor, nomeado por São Francisco, que na crença católica é protetor dos animais, e pela empresa de equipamentos de comunicação. Vem daí também a silhueta do logo.

5.5.6. Experimento adicional

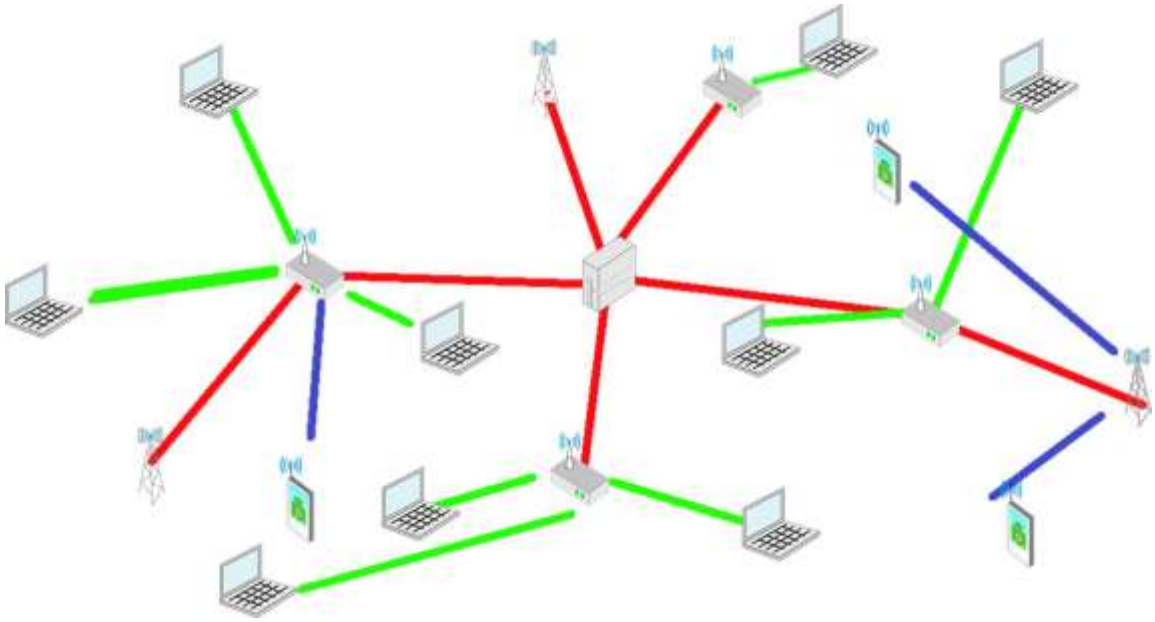
Um experimento adicional foi realizado, porém este sem a comparação com o hardware. Foi realizada uma medição de capacidade com um cenário composto por 12 nós, sendo 3 móveis e 6 fixos, envolvendo enlaces fixos, enlaces sem fio e enlaces móveis. Três células 3G e 4 pontos de acesso foram utilizados, além de uma máquina central provedora de serviços.

Os roteadores estão distantes entre si aproximadamente 1000 metros, mesma distância entre as células 3G. Os nós foram colocados sempre 200 metros próximos a um ponto de acesso ou célula. O servidor foi ligado aos roteadores e as antenas via cabo. Todos os IPs estão na faixa 172.16.0.0/16.

No servidor central foi utilizado o software Apache e um servidor de IRC(ngIRCd[31]), enquanto os clientes utilizaram *wget, lynx* (navegador modo texto) e *irssi* [32](cliente IRC modo texto). O experimento foi executado por meia hora, testando verificando se os clientes eram capazes de acessar uma página hospedada no servidor web, baixar o arquivo de vídeo de 200Mb e trocar mensagens de IRC. Nenhuma operação de *handover* foi estritamente *seamless*, ou seja, não percebida. O *wget* sempre foi interrompido, porém ao retornar a conexão com o outro ponto de acesso, utilizar a opção de continuar reiniciava a transmissão.

O diagrama deste experimento é ilustrado na **Figura 15**.

Figura 15: *Diagrama do experimento completo*



[31] http://ngircd.barton.de/ - acessado em 01/08/2014
[32] http://www.irssi.org/ - acessado em 02/08/2014

5.5.7. Conclusão sobre os experimentos

Em todos os experimentos efetuados, foi possível obter um comportamento próximo ao obtido com o comportamento do hardware, excetuando-se apenas pela menor taxa de transmissão, limitada pelos recursos de entrada e saída.

Com isso, cumprimos o objetivo inicial que era prover um ambiente de experimentação, onde testes efetuados no **NetkitG3** tenham comportamento similar aos experimentos executados em hardware.

Neste momento, está sendo esperado apenas a finalização da **NetGuit**, bem como testes de usabilidade da nova interface, para que o software seja finalizado e enviado aos criados do *Netkit* para avaliação. Independente do resultado e do nome a ser adotado será este, ou outro, os módulos criados, bem como mais detalhes sobre os experimentos executados e os primeiros tutoriais, estão sendo disponibilizados nos sites www.paulogurgel.com.br/mestrado/ e http://www.lsec.icmc.usp.br/gurgel, onde permanecerão disponíveis aos interessados.

5.6. Considerações finais

Este capítulo descreveu o projeto desenvolvido, funcionalidades e módulos de software que foram necessários para a conclusão do projeto. Foram também apresentados estudos de caso que permitiram a verificação e a validação do ambiente desenvolvido.

No próximo capítulo é apresentada a conclusão que inclui as contribuições, as dificuldades encontradas, os trabalhos futuros as publicações científicas oriundas do desenvolvimento deste trabalho.

Capítulo

6

6. Conclusão

A premissa inicial do trabalho era viabilizar a experimentação de redes sem fio com máquinas georreferenciadas, que permitisse realizar a experimentação de cenários de mobilidade e viabilizar a prática de redes sem fio e móveis, seja na tecnologia 802.11, seja na tecnologia 3G.

Uma vez estudadas as ferramentas disponíveis, foi utilizada uma abordagem visando estender uma ferramenta existente, a partir de um conjunto de funcionalidades desejadas inicialmente. É importante relembrar que, sendo uma ferramenta de experimentação, prototipação e estudo, em nenhum momento o projeto se preocupou em reduzir a diferença de desempenho entre as aplicações reais e as experimentais. De fato, não foi introduzido nenhum problema de desempenho na comunicação e a amostragem relativamente lenta não ocupou sensivelmente o processador utilizado nos testes.

A possibilidade de usar o *frontend* gráfico em uma máquina diferente da que executa o *netkitcontrol* também viabiliza a economia de recursos na máquina que executará os experimentos, e este requisito está alinhado a trabalhos futuros.

A comparação entre os experimentos reais e o funcionamento em simulação mostrou que o mesmo comportamento na execução dos testes, embora por vários fatores o desempenho seja diferente, cumpriu os objetivos principais do projeto de pesquisa. Foram mais de 15 módulos de software desenvolvidos, com diferentes funcionalidades e diferentes conhecimentos exigidos, colaborando para a formação profissional e acadêmica do aluno. Com isso, tem-se agora um conjunto de ferramentas que podem ser utilizadas para desenvolver os cenários de rede sem fio.

Até onde foi possível simular em testes, não foi encontrado nenhum cenário de pequena e média escala que não foi possível reproduzir com sucesso. Excluímos deste caso, problemas que vão além da ordem técnica e fogem ao escopo do trabalho, como o *Roaming* que interfere na tarifação dos dados ou questões de comunicação entre operadoras.

A baixa carga permitiu testar estas aplicações em funcionamento, houve desconexões onde as mesmas eram esperadas no mundo real. Também foi possível observar perdas de pacotes e aumento no *delay*, como era esperado ao afastar os *hosts* de seus pontos de acesso.

A partir de agora, será necessário desenvolver atividades acadêmicas que façam bom uso do NetkitG3, considerando o mesmo como um objeto de aprendizagem e desenvolvendo vários tutoriais.

6.1. Dificuldades encontradas

A principal dificuldade encontrada em projetos desta ordem se refere a atualização da documentação. Em muitos projetos *open source*, é comum que as ferramentas de software evoluam mais rapidamente que a documentação, bibliografia e material de apoio disponível. A maior parte da bibliografia para desenvolvedores *Linux* considera versões 2.6 e até mesmo 2.4 do *Kernel*, principalmente quando se considera materiais mais avançados.

Utilizar uma versão de software mais antiga e compatível com o material, embora pudesse simplificar o aprendizado, teria como sacrifício perder as novidades implementadas em versões posteriores. Vários dos recursos de comunicação sem fio foram incluídos em versões relativamente recentes e talvez, portar novidades de comunicação sem fio do *Kernel* para uma versão mais antiga mantendo as interfaces programadas, teria sido um desafio bem maior. Foi considerado também que as mudanças efetuadas no código fonte do *Kernel* e que desatualizaram a documentação, foram necessárias até mesmo para prover tais recursos..

De outro lado, do hospedeiro, também houveram problemas novos causados por modificações na distribuição, como por exemplo, problemas para executar a máquina virtual pelo uso da pasta "/tmp" com o sistema de arquivos "*tmpfs*". Também houve incompatibilidades iniciais com o *udev,* sistema de gerenciamento de dispositivos que cuida do *hotplug* e da conexão de dispositivos, carga e descarga dos módulos.

Modificações de segurança que impedem o funcionamento correto do *Netkit* apareceram algumas vezes, mesmo na versão oficial; dificuldades de executar o laboratório se o mesmo gerar seus arquivos diferenciais no /tmp, ou na pasta do usuário corrente.

Outra dificuldade encontrada foi com a experimentação prática. Diferente de um simulador, ou mesmo de redes cabeadas, o ambiente sem fio é muito instável e a quantidade de fatores externos que influenciam de certa forma na comunicação e muito grande. O mesmo experimento realizado, sem troca de equipamentos ou de local, poderia gerar resultados diferentes por leve diferença na carga de trabalho, pela disposição de carros estacionados nos arredores, quantidade de pessoas presentes e até mesmo o clima.

Embora a transmissão do sinal de satélite com os receptores de GPS seja um padrão bem conhecido, o mesmo não acontece na comunicação entre o GPS e o computador. Diferentes fabricantes possuem protocolos proprietários na troca de informações com suas aplicações, sendo alguns arquivos de texto as exceções. Considerando a ausência deste padrão, esta dificuldade foi contornada com o desenvolvimento de "mais um" protocolo proprietário, bem simples, porém incompatível com aplicações típicas de GPS, caso em que foi necessário desenvolver o cliente GPS.

No caso dos telefones celulares, a implementação também partiu para uma abordagem representativa, pois os softwares das antenas não são facilmente acessíveis, tampouco o equipamento que permita montar uma rede de telefonia privada. Existe uma alternativa chamada OpenBTS, que redesenvolve a pilha GSM (2G) permitindo montar uma rede móvel privada. Todavia, suas funcionalidades são restritas para utilização com VoIP (*Voice over Internet Protocol*) por meio de um servidor Asterisk, divergindo da funcionalidade objetivada no projeto.

Todos os problemas encontrados foram abordados e soluções foram propostas, seja por modificação da estratégia inicial, seja apenas pelo investimento de tempo adicional na execução das tarefas.

6.2. Contribuições

Este trabalho apresentou contribuições para a área de experimentação e ensino de redes sem fio. Com a adoção da ferramenta, professores poderão utilizá-la para aplicar novas atividades práticas aos seus alunos, contribuindo em sua formação.

Aos estudantes, entusiastas e curiosos, o *NetkitG3* proporcionará um ambiente seguro para testes. Não obstante, os cenários de aplicação incluem não somente os cenários básicos de comunicação, mas testes e experimentos de segurança em comunicação sem fio, questões de amplo interesse da comunidade cientifica e empresarial.

As contribuições de software, tão logo estejam devidamente documentadas, serão submetidas aos criadores das ferramentas iniciais, ou seja, as modificações na UML e no *Netkit* serão enviadas aos criadores de tais ferramentas, para possível inclusão na linha principal de desenvolvimento.

No caso do *Netkit*, caso não exista interesse do autor de incluir as ferramentas propostas, por questão de divergir de alguma estratégia ou projeto em andamento, o software resultante será renomeado e disponibilizado como *Open Source* para a comunidade, mantendo as mesmas licenças e a referência ao *Netkit* original. De qualquer modo, os códigos fontes desenvolvidos poderão servir como referência para a criação de atividades e módulos similares.

Será disponibilizado também, no *site* do autor e no *site* do laboratório, um conjunto de tutoriais, que poderão ser utilizados livremente por docentes e estudantes em seus estudos. Neste caso é importante apresentar que em 2013, o site www.paulogurgel.com.br, onde são hospedados os tutoriais do *Netkit* e em breve, do NetkitG3, recebeu 11786 acessos, representando quase 100Gb de transferência de dados. Em 2014, até o dia 11/09/2014, vésperas da entrega desta monografia e última revisão, foram 6612 acessos.

Finalmente, este projeto apresentou contribuições para a formação acadêmica do aluno, no que tange a pesquisa e na colaboração de projetos de iniciação cientifica e trabalhos de conclusão de curso.

6.3. Produção cientifica

Esta seção tem por objetivo apresentar a produção cientifica proveniente dos trabalhos do aluno, bem como de trabalhos de iniciação científica ou trabalhos de conclusão de curso direta ou indiretamente relacionados a esta dissertação.

6.3.1. Participação em Eventos

Membro do comitê de programa do I WOCCES (*Workshop of communications in Critical Embedded Systems*), workshop integrante da 31ª SBRC (Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos), 2013

Membro do comitê de programa do II WOCCES (*Workshop of communications in Critical Embedded Systems*), workshop integrante da 32ª SBRC (Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos), 2014

XXXII Congresso da Sociedade Brasileira de Computação, 2012

XV Semana da Computação – SEMCOMP, 2012

2º Workshop do PAE no Campus de São Carlos, 2012.

3º Workshop do PAE no Campus de São Carlos, 2013.

Sessão de Treinamento sobre *Web of Knowledge*, 2012

2º Workshop de Capacitação de Pesquisadores da USP em Publicação Científica 2012

10º Workshop de Teses e Dissertações, ICMC, 2013.

6.3.2. Produção Intelectual e participação em projetos

Os trabalhos desenvolvidos até o momento, incluindo experimentos, tutoriais e avaliação do uso da ferramenta *Netkit* viabilizaram a obtenção dos seguintes resultados. Um artigo completo e um artigo resumido em conferências internacionais, sendo os mesmos de qualificação B1 e A2 respectivamente, além de 1 artigo completo em congresso nacional. Um aceite de livro completo para publicação (livro já no catálogo, revisões finais e aguardando publicação). Duas participações como colaborador em simpósios de iniciação científica, duas participações como palestrante em minicursos, além de participação em projetos de pesquisa, comitê de programa do Workshop de Comunicação em Sistemas Embarcados Críticos e Workshops.

- GURGEL, P. H. M.; BARBOSA, E. F.; BRANCO, K. R. L. J. C.; Teaching computer networks: a practical approach using virtualization tools, 2013' Frontiers in Education, Oklahoma City, Estados Unidos, Outubro de 2013

- GURGEL, P. H. M.; BRANCO, L. H. C.; BARBOSA, E. F.; BRANCO, K. R. L. J. C.; Development of a practical computer network course through Netkit virtualization tool, 2013' International Conference on Computational Science, Barcelona, Espanha, Procedia Computer Science, Volume 18, 2013, Pages 2583-2586, ISSN 1877-0509

- ROMEIRO, R. F.; BRANCO, K. R. L. J. C.; GURGEL, P. H. M.; - O uso de redes virtuais Netkit no estudo e análise de ataques de negação de serviço distribuídos, XX Congresso de Iniciação Científica, São Carlos – SP, 2013

- GURGEL, P. H. M.; BRANCO, K. R. L. J. C; BRANCO, L. H. C.; BARBOSA, E. F.; TEIXEIRA, M. A. M.; Redes de Computadores – Da Teoria a Prática com Netkit, Livro aprovado para publicação pela editora El Sevier, 2012/2014 (no estágio de pós-produção)

- Participante do projeto de pesquisa: Arquitetura Orientada a Serviços para Sistemas Embarcados Críticos Complexos – Coordenadora: Profa. Dra. Kalinka Regina Lucas Jaquie Castelo Branco – Projeto Regular FAPESP, 2012/2013

- ROMEIRO, R. F.; PIGATTO, D.F.; GURGEL, P. H. M.; BRANCO, K. R. L. J. C., Análise de Ferramentas de Ataques a Redes Ad-Hoc sem fio no contexto de Sistemas Embarcados Críticos. – Apresentação no VI WICT/USP, 2012.

  Nota: Menção Honrosa – Melhor artigo do 6º Workshop de Iniciação Científica e Tecnológica da Universidade de São Paulo, Universidade de São Paulo

- GURGEL, P. H. M. ; BARBOSA, E. F. ; BRANCO, K. C.. A ferramenta Netkit e a virtualização aplicada ao ensino e aprendizagem de redes de computadores. In: XXXII Congresso da Sociedade Brasileira de Computação, 2012, Curitiba. Anais do XXXII Congresso da Sociedade Brasileira de Computação, 2012. (Artigo publicado)

- GURGEL, P. H. M. – Minicurso na 15ª SEMCOMP (Semana da Computação – ICMC/USP) com o título: Redes e Segurança, 2013.

  Nota: Uso do Netkit para demonstrações relacionadas aos temas abordados.

- GURGEL, P. H. M. – Minicurso na 14ª SEMCOMP (Semana da Computação - ICMC/USP) com o título: Introdução à redes de computadores, 2011.

  Nota: Uso do Netkit para demonstrações relacionadas aos temas abordados.

6.4. Trabalhos futuros

Considerando que uma nova ferramenta abre possibilidades para a continuidade do trabalho, algumas sugestões de trabalho que poderão ser executadas na sequência, pelo grupo de pesquisa, pelo próprio autor, ou por leitores que vierem a se interessar pelo assunto no futuro próximo.

1. Desenvolvimento de tutoriais para cobrir os assuntos de redes móveis, sistemas distribuídos em redes móveis e segurança de redes móveis;

2. Instalar e viabilizar o uso do próprio OpenBTS para testes de comunicação VoIP ou, a partir de sua extensão, viabilizar a configuração de uma rede 2G/GSM;

3. Em conjunto com o emulador da Google, viabilizar o uso de máquinas virtuais Android em conjunto com as máquinas virtuais Linux;

4. Aperfeiçoar a interface gráfica **NetGuit**, de modo a permitir que o laboratório seja criado completamente por ele, e que o arquivo de configuração seja gerado pelo mesmo, bem como outros avanços gerais na usabilidade, dispensando o uso de terminais adicionais;

5. Aperfeiçoar o **netkitcontrol** para administrar recursos na criação das instâncias de máquinas virtuais com o propósito de compartilhar instâncias de laboratórios virtuais entre diferentes usuários;

6. Implementar uma ferramenta de apoio a avaliação, auditando alterações em arquivos de configuração, saídas de logs de funcionamento e o histórico de comandos inseridos pelo usuário, de modo a simplificar o procedimento de correção de atividades práticas.

7. Implementar protocolos GPS que viabilizem o uso de aplicações consolidadas.

Referências

ANDERSON, A.; BENEDETTI, R. Head First Networking. 1ª Edição. ed. Sebastopol : O'Reilly, v. I, 2009. ISBN 978-0-596-52155-4.

APACHE, S.F – Apache Web Server – http://httpd.apache.org/ - acessado em 10/09/2014.

BARBOSA, N. M. S.; ANJOS, M. L. D. A.; BOGO, M. Uso do Netkit no Ensino de Roteamento Estático. Anais do XI Encontro de Estudantes de Informática do Tocantins, Palmas, p. 215-222, 2009.

BATTISTA, G. D.; PATRIGNANI, M.; PIZZONIA, M.; RIMONDINI, M. – Netkit-. http://www.netkit.org – acessado em 10/09/2014

BOSOM Holdings. NetSim Cisco Network Simulator 8.0 – http://www.boson.com/netsim-cisco-network-simulator - acessado em 10/09/2014

CARPENTER, T. CWNA Certified Wireless Network Administrator – Oficial Study Guide – Mc Graw Hill – 2008

DIKE, J. User Mode Linux - Prentice Hall – 2006

EVANS, D. H., MCDICKEN, W. N. - Doppler Ultrasound, editora John Wiley and Sons, 2000.

FERNÁNDEZ, D.; GÁLAN, F. VnUml– Virtual Network User Mode Linux – http://neweb.dit.upm.es/vnumlwiki/index.php/Main_Page - acessado em 10/09/2014

FERREIRA, R. E.; Linux, Guia do Administrador do Sistema, editora Novatec, 2003

FUERTES, W. M.; VERGARA, J. E. L. A quantitative comparison of virtual network environments based on performance measurements. EscuelaPolitécnica Superior, Madrid, Spain, 25 Maio 2007. disponível em http://uni-smr.ac.ru/archive/net/NetKit/hpsua07.pdf – acessado em 10/09/2014.

GARG, V. Wireless Communication and Network - Morgan Kaufmann - 2007

GURGEL, P. H. M.; BRANCO, K. R. L. J. C.; – Laboratórios Virtuais no Ensino de Redes de Computadores – Monografia de conclusão de curso – ICMC/USP – 2010

GURGEL, P. H. M.; BARBOSA, E. F.; BRANCO, K. R. L. J. C.; A ferramenta Netkit e a virtualização aplicada ao ensino e aprendizagem de redes de computadores, XX Workshop

sobre Educação em Computação – WEI'2012, XXXII Congresso da Sociedade Brasileira de Computação, CSBC 2012.

HILL, L.; Georeferencing – The Geographic Associations of Information – MIT Press – 2006

IBGE, Glossário Cartográfico, http://www.ibge.gov.br/home/geociencias/cartografia/glossario/glossario_cartografico.shtm, acessado em 10/09/2014

ISC, BIND DNS Server – http://www.isc.org/downloads/bind/ - acessado em 10/09/2014

KUROSE, J. F.; ROSS, K. W. Computer Network: A Top Down Approach. 6Th Edition. Pearson Addison Wesley, 2013.

MAHESWARAN, M.; MALOZEMOFF, A.; NG, D.; LIAO, S.; GU, S.; MANIYMARAN, B.; RAYMOND, J.; SHAIKH, R.; GAO, Y. GINI: A User-level Toolkit for Creating Micro Internets for Teaching & Learning Computer Networking. - 12th ACM SIGCSE Conference on Innovation and Technology in Computer Science Education, March 2009, Chattanooga, Tennessee, USA.

NCO (National Coordination Office for Space-Based Positioning, Navigation, and Timing), Official U.S. Government information about the Global Positioning System (GPS) and related topics, gps.gov – acessado em 10/09/2014

NI, L. M.; ZHENG, P. EMWIN – Emulating a Mobile Wireless Network using a Wired Network - WOWMOM '02 Proceedings of the 5th ACM international workshop on Wireless mobile multimedia – 2002

LIAO, S. Wireless Gini: An emulator for Ad-hoc Wireless Local Area Networks - McGill University – 2006

OPNET Technologies, Inc. OPNET Modeler- http://www.opnet.com/solutions/network_rd/modeler.html - acessado em 10/09/2014

ORACLE. Virtual Box - https://www.virtualbox.org/ - acessado em 10/09/2014

QUOITIN, B. C-BGP- http://c-bgp.sourceforge.net/ - acessado em 10/09/2014

RAYCHAUDHURI, D.; SESKAR, I.; OTT, M.; GANU, S.; RAMACHANDRAN, K.; KREMO, H.; SIRACUSA, R.; LIU, H.; SINGH, M.Overview of the ORBIT Radio Grid Testbed for Evaluation of Next-Generation Wireless Network Protocols - Wireless Communications and Networking Conference 05' – IEEE – 2005.

RIMONDINI, M. Emulation of Computer Networks with Netkit. Università degli Studi di Roma Tre, Roma, 13 jan. 2007. http://www.netkit.org/publications/netkit-tr.pdf - acessado em 10/09/2014.

SANTANA, A. A. Proposta para otimização de desempenho do protocolo TCP em redes wireless 802.11.- Poli USP – 2003

SKLAR, B. Digital Communications Fundamentals & Applications, 2nd Edition, 2005.

STALLINGS, W. Wireless Communication and Network - Prentice Hall - 2nd Edition – 2005

TANENBAUM, A. S. Redes de Computadores. 4ª Edição. ed. São Paulo : Campus, 2003. ISBN 8535211853.

VMWARE Inc. VMWare Player- http://www.vmware.com/products/player/ - acessado em 10/09/2014

XEN PROJECT – Xen – http://www.xenproject.org/ - acessado em 10/09/2014