FUNDAÇÃO DE ENSINO "EURÍPIDES SOARES DA ROCHA"
CENTRO UNIVERSITÁRIO "EURÍPIDES DE MARÍLIA" – UNIVEM

MARIANA MASSIMINO FERES

AVALIAÇÃO DO DESEMPENHO DO PROTOCOLO TCP/IP EM REDES COM TOPOLOGIA ESTRELA UTILIZANDO A FERRAMENTA *NETWORK SIMULATOR*

MARÍLIA
2006

MARIANA MASSIMINO FERES

AVALIAÇÃO DO DESEMPENHO DO PROTOCOLO TCP/IP EM REDES COM TOPOLOGIA ESTRELA UTILIZANDO A FERRAMENTA *NETWORK SIMULATOR*

Monografia apresentada ao Centro Universitário Eurípides de Marília, mantido pela Fundação de Ensino Eurípides Soares da Rocha, como parte dos requisitos para obtenção do Título de Bacharel em Ciência da Computação.

Orientadora:
Kalinka Regina L. Jaquie Castelo Branco.

MARÍLIA
2006

MARIANA MASSIMINO FERES

AVALIAÇÃO DO DESEMPENHO DO PROTOCOLO TCP/IP EM REDES COM TOPOLOGIA ESTRELA UTILIZANDO A FERRAMENTA *NETWORK SIMULATOR*

Banca examinadora da Monografia apresentada ao Centro Universitário Eurípides de Marília, mantido pela Fundação de Ensino Eurípides Soares da Rocha, como parte dos requisitos para obtenção do Título de Bacharel em Ciência da Computação. Área de concentração: Arquitetura de Sistemas Computacionais

.
Resultado: _____

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

1º EXAMINADOR: Profa. Dra. Fátima L. Santos Nunes Marques

2º EXAMINADOR:Prof. Dr. Ildeberto Aparecido Rodello

Marília, 27 de novembro de 2006.

Dedico essa monografia aos meus queridos pais Carmem Maria G. M. Feres e Edgar Feres Filho.

AGRADECIMENTOS

Gostaria de agradecer primeiramente as pessoas que me ajudaram a tornar esse projeto uma realidade: minha querida orientadora Kalinka e ao meu amigo Adriano, sem eles seria difícil a realização desse trabalho.

Tenho o intuito, também, de agradecer aos meus pais, a minha irmã Juliana quem sempre se esforçou para me ajudar, ao Gustavo que esteve comigo durante toda a graduação, minha querida amiga/irmã Bárbara e meus amigos Rodrigo, Danilo e Ana Paula (grupo de estudos fantástico e eficiente).

Desejo a todos muita paz e amor e que Deus ilumine sempre nossos caminhos.

FERES, Mariana M. **Avaliação do Desempenho do Protocolo TCP/IP em Redes com Topologia Estrela Utilizando a Ferramenta *Network Simulator*** . 2006. 87f. Trabalho de conclusão de curso - Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, 2006.

RESUMO

A utilização de redes computacionais tem se disseminado de maneira acentuada devido à necessidade de compartilhamento de recursos e informações. A partir desse contexto, quantidades cada vez maiores de pessoas estão fazendo uso desse mecanismo e assim tornando mais complexa sua construção. Esse cenário torna a simulação de redes uma opção barata e viável. Dessa forma, o presente trabalho realiza a simulação de uma rede com topologia estrela, mais especificamente uma rede LAN, utilizando a ferramenta *Network Simulator* (NS-2). São atribuídos aos agentes injetores de pacotes na rede dois controles de congestionamento distintos, TCP Reno e *Highspeed* TCP, além de diferentes larguras de banda. Dessa forma, é possível observar as respectivas vazões com valores mantidos mais elevados quando utilizado o *Highspeed.*

Palavras-chave: Redes de computadores. Controle de congestionamento. Topologia estrela. Simulação.

FERES, Mariana M. **Avaliação do Desempenho do Protocolo TCP/IP em Redes com Topologia Estrela Utilizando a Ferramenta *Network Simulator*** . 2006. 87f. Trabalho de conclusão de curso - Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha, Marília, 2006.

ABSTRACT

The use of computational networks has spread in way accented due to necessity of sharing of data and information. In this context a lot of people are using this mechanism and becoming complexer its construction. This scene becomes the simulation of nets a cheap and viable option. Of this form, the resent work carries through the simulation of a net with topology star, more specifically a net LAN, using the tool Network Simulator (NS-2). Two distinct controls of congestion, TCP Reno and Highspeed TCP are attributed to the injector agents of packages in the net, beyond different widths of band. Of this form, it is possible to observe the respective outflows with kept values more raised when used the Highspeed.

Keywords: Network computers. Congestion control. Star Topology. Simulation.

LISTA DE ILUSTRAÇÕES

LISTA DE SIGLAS

ABNT: Associação Brasileira de Normas Técnicas

ACK: *Acknowledged*

ANSI: *American National Standards Institute*

CBR: *Constant Bit Rate*

DNS: *Domain Name System*

FAST TCP: *Fast Active-queue-management Scalable TCP*

FDDI: *Fiber Distributed Data Interface*

FTP: *File Transfer Protocol*

GCC: GNU C *Compiler*

HDLC: *High-Level Data Link Control*

HTTP: *Hyper Text Transfer Protocol*

HSTCP: *HighSpeed* TCP

IP: *Internet Protocol*

ISO: *Organization for Standardization*

LANs: *Local Area Network*

MAC: *Medium Access Control*

MAN: *Metropolitan Area Network*

MSS: *Maximums Size Segment*

PARSEC: *PARallel Simulation Environment for Complex systems*

OTCL: *Object-oriented Tool Command Language*

RFC: *Request for Comments*

RM-OSI: *Reference Model for Open Systems Interconnection*

SDLC: *Synchronous Dara Link Control*

SMTP: *Simple Mail Transfer Protocol*

STCP: *Scalable TCP*

TCP: *Transmission Control Protocol*

UDP: *User Datagrama Protocpl*

VBR: *Variable Bit Rate*

XCP*: Explicit Congestion Control Protocol*

WANs: *Wide Area Network*

SUMÁRIO

INTRODUÇÃO

Com a evolução das redes de computadores surgiu a necessidade de se estipular regras para a comutação de informações entre os diversos equipamentos interligados, de modo que todos sigam o mesmo conjunto de especificações com a finalidade de estabelecer uma linguagem única para a comunicação. O *software* de comunicação básico exigido para dar suporte às aplicações distribuídas utiliza, para sua implementação, uma estrutura modular, conhecida como arquitetura de protocolos (STALLINGS, 2005).

Duas arquiteturas serviram como base para o desenvolvimento de padrões de protocolos interoperáveis: o conjunto de protocolos TCP/IP, o mais utilizado, e o modelo de referência OSI. (STALLINGS, 2005).

A arquitetura TCP/IP enfatiza a interligação de diferentes tecnologias de rede, tendo como base a idéia de não existir uma tecnologia específica que atenda às necessidades dos usuários, portanto para possibilitar a troca de um alto volume de informação entre diversas pessoas deve-se interligar as redes às quais elas estão conectadas, formando uma inter-rede (COLCHER *et al.*, 2004).

O controle de congestionamento é uma das tarefas realizadas pelos protocolos da camada de transporte, entre eles o TCP. Porém o controle adotado pelo protocolo citado não é aplicável a redes *Gigabits*, uma vez que limita a janela de congestionamento em valores a baixo daqueles alcançáveis por esse tipo de interligação (REZENDE, 2005). Para suprir essa necessidade, vários protocolos de transporte direcionados para rede de alta velocidade estão surgindo, entre eles, o *HighSpeed* TCP (HSTCP), uma versão modificada do TCP proposta por Sally Floyd (REZENDE, 2005).

Além da arquitetura de protocolos, é necessário definir a topologia a ser utilizada, ou seja, qual a maneira mais eficiente de dispor os enlaces físicos e os nós de comutação,

determinando os caminhos físicos existentes e acessíveis entre pares de estações interconectadas (COLCHER *et al.*, 2004). São conhecidas diversas topologias (ponto-a-ponto, anel, barramento, totalmente ligada, parcialmente ligada, entre outras), porém o enfoque deste trabalho é na topologia estrela, apresentada como a interligação de um número limitado de nós a um nó central (concentrador), por meio do qual todo o tráfego da rede e todas as decisões de direcionamento das mensagens são gerenciados. Esse tipo de arranjo permite uma maior disciplina na rede, garante a ocorrência do tráfego, diminui o tempo de espera, possui maior facilidade de instalação, entre outros benefícios. Embora possua todas as qualidades descritas, a topologia estrela apresenta uma desvantagem considerável observada no comutador, uma vez que este pode tornar-se um ponto de falha.

O aumento da complexidade das redes de computadores, aliado à necessidade de acoplar, com uma freqüência cada vez maior, mais elementos a ela, tornou o uso dos simuladores uma estratégia viável e eficaz.

Existem na literatura diversos simuladores tais como: *Network Simulatror (*NS) (FALL, 2006), QNET (QNET, 2004), CNet (CNET, 2002) e SSFNET (SSFNET, 2004), os quais têm como objetivo simular redes de computadores. Contudo, um dos simuladores mais utilizados é o NS, o qual oferece suporte à simulação de um grande número de tecnologias de rede (com ou sem fio) (COUTINHO, 2003).

Com foco nas características citadas, o presente projeto avalia o desempenho do protocolo TCP por meio do TCP Reno e *Highspeed* TCP, auxiliado pelo simulador de redes NS-2. A disposição dos nós é em forma de estrela, caracterizado por uma rede local (LAN).

A monografia apresenta primeiramente a revisão bibliográfica, com conceitos relacionados à redes de computadores (Capítulo 1), arquitetura de protocolos TCP/IP (Capítulo 2) e simulação (Capítulo 3). Em seguida, são mostrados os resultados, bem como a

metodologia usada para obtenção dos mesmos. Por fim, as devidas conclusões são apresentadas. .

1. INTRODUÇÃO A REDES DE COMPUTADORES

Este Capítulo tem como objetivo explorar os conceitos básicos sobre redes de computados. Serão abordados temas como histórico da comunicação, uso das redes de computadores, *hardware* de rede e *software* de rede.

A seqüência de abordagem desse Capítulo teve embasamento, principalmente, nos conceitos apresentados por Tanenbaum (2003), entretanto alguns tópicos são acrescidos de informações extraídas de outros autores.

1.1. Histórico da comunicação

A necessidade de comunicação entre os seres humanos tornou possível o desenvolvimento de várias técnicas (desde sinais de fumaça até a invenção da Internet), as quais permitiam a comutação de informação em distâncias cada vez maiores. Desde os sinais de fumaça até a invenção da Internet o homem vem procurando meios para viabilizar a troca de mensagens.

A comunicação por meio de sinais elétricos teve seu primórdio na invenção do telégrafo pelo engenheiro francês Claude Chappe (CABRITA, 2001), instrumento gerador de pulsos elétricos (código Morce) transmitidos e traduzidos por um operador. Desde então, esta forma de transmissão passou por uma grande evolução, dando origem a grande parte dos aparelhos encontrados atualmente (COLCHER *et. al.*, 1995).

Contudo a evolução no tratamento da informação não ocorreu apenas nas técnicas de transmissão dos dados, mas também nas formas como são armazenados. A conseqüência da união dessas tecnologias - processamento e transmissão de dados – abriram as fronteiras para desenvolvimento de sistemas computacionais mais poderosos e eficazes.

A interconexão de sistemas computacionais veio atender duas necessidades distintas: confecção de sistemas mais confiáveis e maior desempenho e compartilhamento de recursos. A primeira é taxada, por alguns autores, como objetivos de um *Sistema Distribuído*, enquanto que a segunda é tida como uma finalidade de *Redes de Computadores.*

Existe na literatura uma grande divergência entre os conceitos de redes de computadores e sistemas distribuídos. A principal diferença entre essas tecnologias é a maneira como a disposição dos computadores é apresentada ao usuário. Na primeira, é conhecido o destino e a origem das mensagens enviadas, porém na segunda existe transparência na execução dos programas, cabendo ao sistema operacional escolher o elemento de processamento, localizar e transportar todos os arquivos de entrada necessários e direcionar os resultados para os devidos lugares (TANENBAUM, 1997).

Uma rede de computadores pode ser definida como sendo um conjunto de elementos de processamento interconectados por meio de um sistema de comunicação permitindo o compartilhamento de recursos computacionais e a troca de informações via regras consistidas em protocolos (Figura 1). O sistema de comunicação é constituído da interligação de vários módulos processadores por meio de enlaces físicos somado a um conjunto de regras para organizar a comunicação.

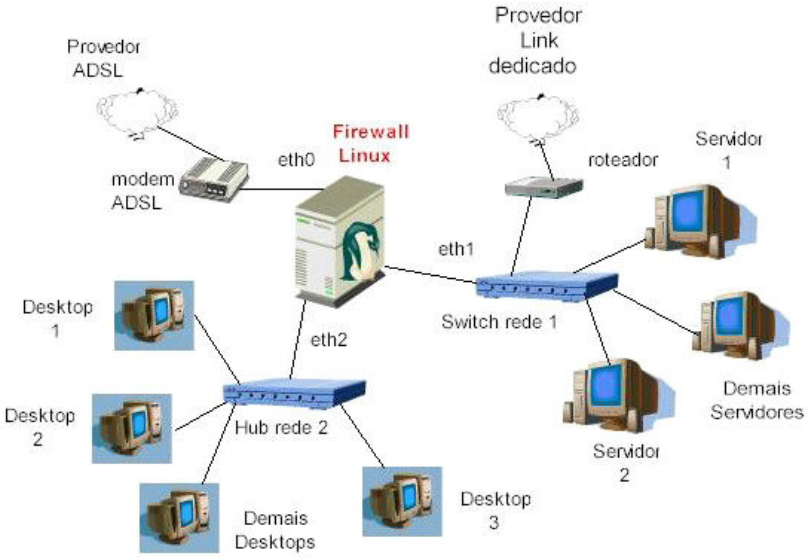

Figura 1: Representação de redes de computadores
Fonte: http://www.antunes.eti.br/ADSL/Rede1.jpg

1.2. Utilização das redes de computadores

A necessidade de comunicação torna vasto o campo de atuação das redes computacionais. São inúmeras as áreas onde essa tecnologia exerce influência, as quais podem ser divididas em: Redes Corporativas e Pessoais.

Porém, com a implantação em larga escala de redes, criam-se problemas sociais, éticos e políticos, difíceis de serem ignorados.

A necessidade de interligar programas, equipamentos e especialmente dados nas corporações, aliada à necessidade de replicação de dados para aumentar a confiabilidade, torna imprescindível o uso das redes, com objetivo principal compartilhar de recursos.

Outro fator relevante na utilização de redes é o custo/desempenho dos pequenos computadores em relação aos computadores de grande porte. Em outras palavras, os

mainframes possuem poder computacional maior que computadores convencionais, porém seu custo é elevado.

Dessa forma, muitos projetistas desenvolveram sistemas baseados em computadores pessoais com um ou mais servidores de arquivos para armazenar os dados. Essa arquitetura é denominada modelo cliente/servidor, ilustrada na Figura 2.

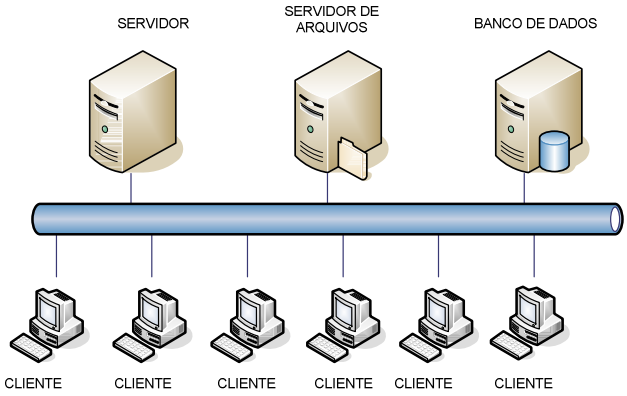

Figura 2: Modelo cliente/servidor tradicional

Pode-se, também, acoplar mais elementos ao sistema para aumentar o desempenho à medida que cresce o volume de carga (escalabilidade). Em sistemas centralizados, quando o poder de processamento torna-se inadequado para a demanda de aplicativos, é necessário modificar o sistema todo. Com o uso do modelo cliente/servidor é possível incluir novos clientes e novos servidores na rede de acordo com a necessidade.

Externo à tecnologia, encontra-se outra vantagem importante no uso de redes: comunicação entre as pessoas. Os funcionários de uma empresa, ou os professores e alunos de uma instituição podem comunicar-se por meio da rede de maneira eficaz com possibilidade de escreverem relatórios simultaneamente, visualizarem dados alterados instantaneamente, entre outras facilidades.

Foram listados, até agora, apenas vantagens tecnológicas e econômicas vinculadas à instalação de redes de computadores. Porém existe outra vertente importante – redes para pessoas.

As redes pessoais começaram a oferecer serviços a partir de 1990 com a proposta de realização de três possíveis serviços:

- Acesso remoto de informações, representado pelas diversas opções que o usuário possui de acesso a bases de dados remotas, por exemplo, *sites* de notícias *online*, ou mesmo de transações financeiras.
- Comunicação interpessoal, necessária para aproximar pessoas separadas geograficamente, permitindo o inter-relacionamento entre elas por meio de facilidades tais como e-mail e videoconferência.
- Diversão interativa contida nos vídeos *online* e, mais ainda, nos *games* que a cada dia ocupam mais mercado.

1.2.1. Questões Sociais

A implantação em larga escala de redes de computadores origina uma série de problemas relacionados ao conteúdo das mensagens que trafegam nessa rede.

Conteúdos polêmicos e indevidos são transmitidos e causam conflitos em relação à proibição ou não dessas informações. Há quem defenda a idéia de que cada pessoa é responsável pelo que acessa e explora, porém deve-se levar em consideração que cada vez mais crianças e adolescentes estão tendo acesso aos dados disponíveis, os quais exercem uma considerável influência em suas formações.

Outra área polêmica envolve direitos de empregadores e empregados. Alguns empregadores acreditam ter o direito de acessar as mensagens de seus funcionários escritas em ambiente de trabalho, porém nem todos os empregas concordam com essa atitude.

Exportando esse princípio para um nicho maior, tem-se a vigia de mensagens por parte de organizações governamentais não apenas de empregados, mas de uma nação inteira (TANENBAUM, 2003).

Em resumo, os problemas envolvidos na utilização de redes de computadores estão longe de serem resolvidos, partindo do princípio de que envolvem tipos variados de pessoas com inclinações e necessidades diferentes.

1.3. Hardware de rede

Além das questões econômicas e sociais envolvidas na implantação de redes, tem-se a parte estrutural, entretanto, não existe uma taxonomia que as classifiquem. Duas estruturas fundamentais são necessárias para sua construção: sistema de comunicação e tecnologia de transmissão (TANENBAUM, 2003).

Link de difusão é uma das duas tecnologias de transmissão usualmente disseminadas. Constitui-se de apenas um canal de comunicação para ser usado por todas as máquinas da rede. Os fragmentos das mensagens, muitas vezes chamados de pacotes, emitidos por qualquer máquina, são recebidos por todas as outras, e caso a mensagem recebida não seja destinada para o nó, ela simplesmente é descartada.

Na maioria das vezes, as redes de difusão permitem o envio de mensagens a todas as máquinas da rede através da utilização de um código especial no campo de endereçamento. Essa forma de dispersão da mensagem é chamada de *broadcasting* ou difusão. Existem, também, sistemas onde a mensagem pode ser destinada apenas para um subgrupo das máquinas da rede, esse mecanismo é conhecido como *multicasting* ou multidifusão.

Em contraste, as redes ponto-a-ponto compreendem muitas ligações entre pares individuais de máquinas. Casualmente, um pacote pode passar por nós intermediários para chegar ao destino, pois, geralmente, esse tipo de arranjo é utilizado para interligações maiores, enquanto que a forma de conexão por difusão é utilizada em redes menores em termos geográficos. Transmissões ponto-a-ponto são conhecidas às vezes como *unicasting*.

O sistema de comunicação é constituído de um arranjo topológico interligando as máquinas por via de enlaces físicos (meios de transmissão) juntamente com um conjunto de regras que organizam a comunicação (COLCHER *et. al.*, 2004).

Redes nas quais os módulos processadores encontram-se menos afastados são ditas confinadas, e sistemas em que os computadores encontram-se separados por alguns metros são chamados Redes Locais de Computadores. Quando essa distância aumenta, podendo chegar às cidades vizinhas, as redes passam a serem chamadas de Redes Metropolitanas, e em nível de país e continente, Redes Geograficamente Distribuídas.

1.3.1. Redes Locais (LAN)

As Redes Locais ou LANs (*Local Área Networks*) são redes privadas contidas em um único estabelecimento ou instituição, podendo ter alguns metros de extensão. Essa é uma rede que permite a interconexão de dados em uma pequena região (distância entre 100 m e 25 Km) (COLCHER *et. al.*, 2004). Seu uso é amplo devido às facilidades que proporciona, tais como o compartilhamento de recursos (impressora, por exemplo) e a troca de informações.

Existem três fatores que distinguem as LANs de outros tipos de redes: tamanho, tecnologia de transmissão e a topologia. Com seu tamanho reduzido, é possível obter, com

antecedência, qual o pior tempo de transmissão e, conseqüentemente, saber qual o melhor arranjo para uma rede específica, além de ajudar na gerência de rede. Outras características típicas associadas às redes locais é a alta taxa de transmissão e a baixa taxa de erro, porém é viável lembrar que essas características compreendem as tecnologias atuais as quais estão, constantemente, sujeitas a evoluções. Tradicionalmente utilizam tecnologia de transmissão *broadcast* ou difusão.

As LANs de difusão admitem várias topologias tais como anel, estrela e barramento. Em uma rede de barramento todas as estações ligam-se ao mesmo meio físico de transmissão configurando uma tecnologia multiponto (COLCHER *et. al.*, 2004). Comumente cada nó conectado à barra tem acesso às informações transmitidas. Esta característica facilita o uso do *broadcast* e *multicast*. Caso seja desejado que mais de uma máquina envie mensagens, é necessário um mecanismo de arbitragem, seja ele centralizado ou distribuído, como, por exemplo, a configuração *Ethernet*, uma rede de difusão com barramento descentralizado representada pelo padrão IEEE 802.3 (TANENBAUM, 2003).

A simplicidade e o custo reduzido de instalação justificam o uso desse tipo de disposição de uma rede de computadores. Em contraposição, pode haver quebra de um único cabo ou o mau funcionamento de um nó, o que irá interferir em toda a rede.

Topologia em anel consiste em estações conectadas através de um caminho fechado, formando um círculo (COLCHER *et. al.*, 2004). Cada *bit* se propaga de modo independente sem ter que esperar pelo pacote ao qual pertence. As estações não são interligadas diretamente ao meio de transmissão, utilizam uma série de repetidores ligados fisicamente por motivos de confiabilidade.

Em geral, as redes anel possuem transmissão unidirecional (embora possuam capacidade de transmitir dados em qualquer direção) a fim de simplificar o projeto dos repetidores e reduzir a complexidade do protocolo de comunicação que assegurará a entrega

correta dos pacotes. Existem na literatura algumas configurações de redes em anel tais como a rede *Token Ring* da IBM, especificada pelo padrão IEEE 802.5 (COLCHER *et al.*, 2004), e o FDDI (*Fiber Distributed Data Interface*), constituída por dois anéis de fibra ótica com taxa de transmissão de 100 Mbps, conectados em cada computador. É utilizado para recuperação automática de falhas (COMER, 2005).

Outra forma de dispor os nós de uma rede computacional é em forma de estrela. Esse tipo de arranjo recebe o nome de topologia estrela e terá a sessão 2.4.1.1 dedicada à sua explicação devido à sua importância no contexto desse trabalho, além de seu uso disseminado em empresas e instituições.

1.3.1.1. Topologia Estrela

A maior característica no arranjo dos elementos de processamento em forma de estrela é a presença de um elemento central utilizado para gerenciar o fluxo de dados. Cada nó liga-se, com tecnologia ponto-a-ponto, ao elemento central. Dessa forma todo dado enviado deverá passar pela central obrigatoriamente, o que agiliza a transmissão (comparado aos outros modelos apresentados), uma vez que os dados não irão passar por todas as estações. O concentrador encarrega-se de encaminhar o sinal para as estações solicitadas, economizando tempo. Nada impede que haja comunicação simultânea, desde que as estações envolvidas sejam diferentes.

Essa disposição dos nós é utilizada tipicamente em redes de computadores em que as estações secundárias alimentam um nó central processador de informações, porém uma

situação comum é a utilização desse nó de forma a gerenciar a comunicação e a operação de diagnóstico.

O gerenciamento realizado pelo elemento central pode ser por *chaveamento de pacotes* ou *chaveamento de circuitos*, operando em *modo de transferência assíncrona* (ATM – *Asynchronous Transfer Mode*) (COLCHER *et al*., 2004). No chaveamento por pacotes, os pacotes são enviados do nó origem ao elemento central de processamento e em um momento propício é transmitido para o nó destino. Já o chaveamento de circuitos estabelece uma conexão entre os nós origem e destino, a qual persiste durante toda a comunicação. Dessa forma fica barrada a comunicação de qualquer um desses nós participantes da conexão com outro elemento da rede durante a participação.

Redes CBX (*Computerized Branch Exchange*), chaveamento computadorizado, são exemplos de chaveamento por circuito, sendo que este é realizado por um PABX (*Private Automatic Branch Exchange*). Embora possua arquitetura e tecnologia diferentes das demais redes locais, o CBX foi incluído nessa categoria por ser uma alternativa para interligação de dispositivos digitais.

A topologia estrela é caracterizada por um elemento central que "gerencia" o fluxo de dados da rede, estando diretamente conectado (ponto-a-ponto) a cada nó, o que deu origem à designação "Estrela". Os agregados computacionais (*Clusters*), em geral, podem ser interligados em forma de estrela com o auxílio de um equipamento de rede local (*hub* e *switch* ou mais atualmente *hub-switch*) o qual representa o dispositivo de interconexão. Os inúmeros computadores compartilham um único meio para se comunicarem, a rede local (DANTAS, 2005).

1.3.2. Redes Metropolitanas (MAN)

Segundo Comer (2005) as redes metropolitanas compreendem qualquer tecnologia nova de rede física atuante em distâncias grandes o suficiente para uma área metropolitana com taxas de transmissão por volta de centenas de *megabits* por segundo até de vários *gigabits* por segundo.

Uma rede metropolitana ou MAN abrange uma cidade. Essa configuração de rede começou a ser utilizada a partir de sistemas de televisão com antenas comunitárias usados em áreas com baixa recepção de sinais pelo ar e evoluiu para sistemas de televisão a cabo. Com a obtenção de concessões dos governos municipais, as empresas ficaram aptas a interligarem cidades inteiras.

Com o constante crescimento da Internet, as operadoras de redes de televisão a cabo perceberam a possibilidade de oferecer serviços da Internet em partes do espectro não utilizadas com apenas algumas modificações no sistema, ou seja começaram a utilizar faixas de freqüências não aproveitadas na transmissão televisiva (TANENBAUM, 2003).

1.3.3. Redes Geograficamente Distribuídas (WAN)

Redes de comutação de pacotes que operam em grandes distâncias possuem diferenças básicas das redes locais. Uma rede WAN (*Wide Area Network*), ou de longa distância, possibilita a comunicação em nível internacional ou intercontinental. A maioria das tecnologias de rede de longa distância não determina distâncias efetivas para a interligação.

Geralmente operam em velocidades mais lentas e necessitam de um tempo de retardo maior (COMER, 2005).

As WANs são constituídas por um conjunto de máquinas (*hosts*) com a finalidade de executar aplicativos do usuário (TANENBAUM, 2004). Essas máquinas são interconectadas por uma sub-rede de comunicação pertencente a um provedor de serviços da Internet. Essa sub-rede tem como finalidade transportar uma mensagem de um *host* para outro por meio de dois componentes: linhas de transmissão e elementos de comutação.

As linhas de transmissão podem ser enlaces de rádio, fios de cobre ou fibra óptica e são utilizadas para transportar os *bits* entre as máquinas. Já os elementos de comutação são equipamentos especializados, os quais encaminham os dados que chegam a uma linha de entrada para uma linha de saída. Atualmente esses elementos de comutação recebem o nome de roteador.

O princípio de funcionamento de uma WAN é basicamente o encaminhamento das mensagens entre os roteadores da rede. Caso um roteador queira comunicar-se com outro sem possuir uma linha de comunicação ele terá que realizar essa transmissão por via de outros roteadores. Cada comutador intermediário recebe um pacote e o armazena até que uma linha de saída seja liberada para seu encaminhamento. Esse princípio nomeia a sub-rede que o utiliza como sub-rede de *stored-and-forword* – de armazenamento e encaminhamento – ou comutação de pacotes (TANENBAUM, 2004).

Além das características enumeradas acima, os roteadores, como o próprio nome sugere, tem a finalidade de escolher a melhor rota a ser seguida pelo pacote. Para essa definição de rota são utilizados algoritmos de roteamento, tais como roteamento pelo caminho mais curto, inundação, com vetor de distância, entre outros.

Outra configuração para redes geograficamente distribuídas é a utilização de satélites, não comutada por pacotes. Cada roteador tem a possibilidade de receber e enviar para o

satélite. Esse tipo de disposição é mais adequado para redes onde a propriedade de difusão é importante, uma vez que são inerentemente redes de difusão.

1.4. Redes Sem Fio

De acordo com Colcher (2004) redes sem fio compreendem transmissão de pacotes por canal aberto (“pelo ar”), em canais de freqüência infravermelha ou rádio (radiodifusão), sendo a última a mais difundida atualmente.

Geralmente as redes sem fio oferecem suporte às redes de fiação com o intuito de disponibilizar o acesso a arquivos, base de dados e à Internet. Uma antena capta o sinal de rádio e, por via de uma rede cabeada distribui o acesso aos demais computadores.

A utilização das redes sem fio torna-se cada vez maior, com o desenvolvimento de novas tecnologias e disseminação entre as empresas. Como exemplo, as redes Wi-Fi (*Wireless Fidelity*) cobrem tanto os escritórios da sede da *General Motors* em São Caetano, na Grande São, Paulo, como o chão de fábrica (INFO, 2005).

A disputa pela transmissão sem fio está ganhando proporções significativas, com o surgimento de novas tecnologias. A mais disseminada é, sem dúvida a família 802.11, porém novas especificações de padrões estão surgindo como o IEEE 802.16e redes WiMax (*Worldwide Interoperability for Microwave Acess*) que traz um importante componente à rede: acessa dados em movimento e redes UWB (*Ultra Wideband*) com alcance de cerca de dez metros especificado pelo padrão 802.15.3, além disso tem como objetivo interligar à rede dispositivos domésticos como DVDs (INFO, 2005).

1.5. *Software* de Rede

Nos primeiros projetos de redes de computadores a grande preocupação era o *hardware*, deixando o *software* como artigo secundário. Porém, atualmente, esse modelo foi abandonado e o *software* passou a ser o principal foco para providenciar a comunicação (TANENBAUM, 2003).

A estruturação desse *software* é feita em camadas ou níveis sobrepostos, os quais oferecem serviços para a camada superior por meio de interfaces. Dessa forma a implementação dos níveis é escondida de quem os usa.

O diálogo entre camadas de máquinas diferentes (comunicação entre entidades pares) é permitido pelo protocolo da camada, o qual estabelece as regras para que a comunicação ocorra. Os dados não são transferidos de uma entidade para outra diretamente (de maneira horizontal, como mostram as linhas pontilhadas na Figura 3), são passados à camada imediatamente abaixo, recebendo a cada estágio, informações de controle. Após ultrapassar a camada 1, encontra-se o meio físico onde ocorre, efetivamente, a comunicação (comunicação vertical). Para que as camadas pares adjacentes se comuniquem corretamente, é necessário estabelecer interfaces bem definidas, de maneira a representar claramente os serviços oferecidos por cada camada.

Um conjunto de camadas e protocolos é chamado de arquitetura de protocolos (TANENBAUM, 2003). Os detalhes da implementação e a especificação das interfaces não pertencem à arquitetura. Eles permanecem no interior de cada máquina, de forma a oferecer serviços transparentemente. A especificação da arquitetura de protocolos é suficiente para o desenvolvedor implementar o programa ou construir o *hardware* de cada camada para obedecer ao protocolo adequadamente.

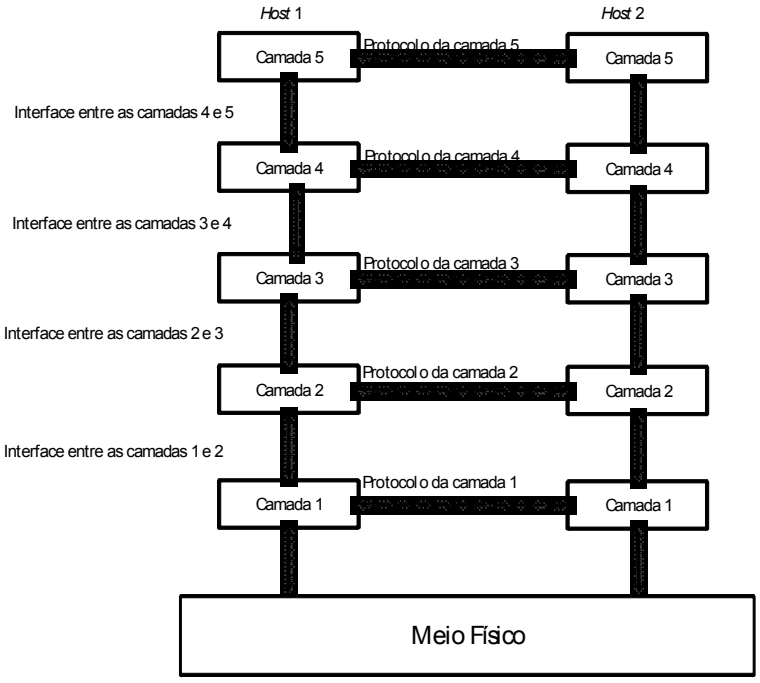

Figura 3: Representação vertical e horizontal da comunicação entre camadas.

As camadas podem oferecer dois tipos de serviços às camadas acima: orientado a conexão e sem conexão. O primeiro exige o pré-estabelecimento da conexão antes de transmitir os dados efetivamente. Já no serviço sem conexão, a mensagem carrega o endereço de destino, podendo seguir por qualquer caminho possível (TANENBAUM, 2003).

Em princípio cada fabricante desenvolvia sua própria arquitetura, a fim de interligar seus computadores, constituindo as chamadas arquiteturas proprietárias (COLCHER *et al*., 2004). Porém a maioria das organizações mantinha um parque de computadores compostos por diversos fornecedores diferentes e, para permitir o intercâmbio de informações, foi preciso definir uma arquitetura única e pública (para não prejudicar nenhum fabricante).

É nesse cenário que a *International Organization for Standardization* (ISO) definiu o modelo *Reference Model for Open Systems Interconection* (OSI) constituído de sete níveis como referência para a construção de arquitetura de protocolos de redes computacionais (COLCHER *et al*., 2004).

1.6. Modelo OSI da ISO

A ISO é uma organização internacional fundada em 1946 com o intuito de elaborar padrões internacionais. Seus membros são os órgãos de padronização de 89 países, dentre eles a ABNT, representante do Brasil, e o ANSI, dos Estados Unidos (COLCHER *et al.,* 2004).

O modelo OSI, criado em 1977 pela ISO, tem como objetivo definir padrões de conectividade para a interligação de redes de computadores (CYCLADES BRASIL, 2001) e constituiu o primeiro passo em busca da padronização dos protocolos embutidos nas camadas. É raramente utilizado comercialmente, sendo considerado modelo *de jure*, pois o modelo é bastante geral e válido, apesar de suas características serem relevantes (TANENBAUM, 2003).

Foram estabelecidas sete camadas funcionais para facilitar a compreensão das questões de projeto sobre um protocolo de comunicação entre dois processos em uma rede. São, em ordem crescente, a camada física, de enlace, de rede, de transporte, de sessão, de apresentação e de aplicação.

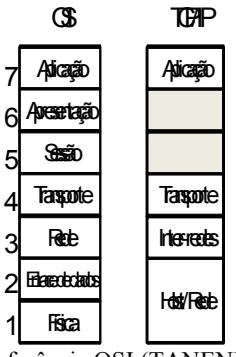

Figura 4: O modelo de referência OSI (TANENBAUM, 2003).

A camada física é responsável pelas especificações de *hardware*, as quais compreendem aspectos mecânicos, elétricos e físicos. Trata como será a representação dos *bits* 0 e 1 na comunicação, de forma que as duas máquinas comunicantes reconheçam esses

bits da maneira como foram enviados. Sendo assim, cabe ao projetista definir a representação e o intervalo de sinalização de cada *bit*, a forma de transmissão (*half-duplex* ou *full-duplex*), como será estabelecida e desfeita a comunicação, a quantidade e significado dos pinos presentes no conector, além de outros detalhes técnicos e mecânicos. A função principal dessa camada é permitir o tráfego de uma cadeia de bits sem a preocupação com seu significado ou forma (COLCHER *et al*., 2004).

Posteriormente à camada física encontra-se a de enlace de dados. Nessa etapa os dados são divididos em quadros com a finalidade de detectar e opcionalmente corrigir eventuais erros que ocorram na camada anterior. Outro aspecto tratado pela camada de enlace é o controle de fluxo da comunicação para evitar que o emissor envie dados mais rapidamente que o receptor possa processar. Muitas vezes o controle de fluxo e a detecção de erros estão integrados (TANENBAUM, 2003). Como exemplo de protocolo desta camada tem-se o HDLC (*High-level Data Link Control*), protocolo orientado a *bit* derivado do SDLC (*Synchronous Data Link Control*) da IBM (TANENBAUM, 2003).

Na seqüência aparece a camada de rede. Nela o conhecimento sobre a disposição dos nós passa a existir. Sua principal finalidade é controlar a operação de sub-rede através da definição de rotas para os pacotes e controle de possíveis gargalos. Existem duas filosofias de acordo com o serviço de rede oferecido pela camada: circuito virtual e datagrama (COLCHER *et al*., 2004). No primeiro é necessário estabelecer uma conexão antes de, efetivamente, enviar os pacotes (orientado à conexão). Já no modelo datagrama, os pacotes não possuem dependências com quaisquer outros pacotes que tenham sido ou serão enviados (serviço não orientado à conexão). Nesse tipo de serviço, o roteamento é calculado a cada pacote encaminhado por um nó da rede.

Logo após a camada de rede tem-se a de transporte. Caracterizada por estabelecer uma comunicação fim a fim, não presente nos níveis descritos anteriormente, essa camada é

responsável pela multiplexação (várias conexões de transporte partilhando uma mesma conexão de rede) e pelo fracionamento (uma conexão de transporte ligada a várias conexões de rede), realiza controle de fluxo e é apta para detectar e corrigir determinados erros fim a fim. Tem a função, também, de segmentação da mensagem em outras menores.

A próxima camada na hierarquia do modelo OSI é a camada de sessão. Esta propicia, aos usuários de máquinas distintas, o estabelecimento de sessões entre si. Uma sessão oferece diversos serviços tais como o gerenciamento de *token* (apenas o proprietário do *token* pode transmitir seus dados), o controle de diálogo através de marcas lógicas (pontos de sincronismo) e o gerenciamento de atividades.

Uma das últimas camadas do modelo do protocolo descrito nessa sessão é a camada de apresentação. A camada preocupa-se, não mais com manipulação de *bits*, mas sim com a semântica e sintaxe das informações a serem transmitidas. Dessa forma é necessário conhecer a sintaxe do sistema local, bem como do sistema de transferência.

Por fim, a camada de aplicação é a última a ser especificada pelo modelo. Ela fornece os meios para as aplicações utilizarem o ambiente de comunicação OSI. Contém uma série de protocolos utilizados pelos usuários, como por exemplo, o HTTP (*HyperText Transfer Protocol*), base da *web.*

1.7. Considerações Finais

A partir da necessidade de interligação entre elementos de processamento e seguindo o modelo de referência OSI, outro protocolo de comunicação surgiu e revolucionou o

mecanismo para toca de informações. Essa nova arquitetura foi chamada de TCP/IP devido aos dois principais protocolos e será apresentada no próximo capítulo.

2. ARQUITETURAS DE PROTOCOLOS TCP/IP

O presente capítulo tem como objetivo explorar características do protocolo de comunicação *Tansfer Control Protocol / Internet Protocol* (TCP/IP), bem como sua estrutura e utilização. As sessões estão divididas em histórico e organização, com subseções nesta última, as quais listam as diversas funcionalidades de cada camada, dando ênfase aos protocolos TCP e IP.

2.1. Histórico

Ao longo dos anos, agências norte-americanas vêm financiando pesquisas em torno da interconexão de computadores, mais precisamente na área de interconexão global de redes (COMER, 2005).

Tendo em vista esse aspecto, Stallings (2005) contextualiza o surgimento do protocolo TCP/IP como sendo o resultado das pesquisas realizadas pela *Advanced Research Projects Agency* (ARPA) sobre a rede experimental de comutação de pacotes ARPANET.

Em meados da década de 1970 iniciaram-se as primeiras atividades da ARPA focando a interligação em redes, e a arquitetura atual somente foi estabelecida entre 1977 e 1979 (COMER, 2005). Nesse escopo, a agência norte-americana foi pioneira no financiamento de pesquisas de comutação de pacotes.

A ARPA é um órgão do departamento de defesa dos Estados Unidos (*Department of Defense* - DoD), cuja tecnologia comporta um conjunto de padrões, para especificar os

detalhes do sistema de comunicação, e um conjunto de convenções para roteamento e interconexão em rede. Aos poucos, diversos órgãos foram conectados à ARPANET através de linhas telefônicas dedicadas.

Com o surgimento das redes de rádio e satélite, houve a necessidade de especificar uma nova arquitetura de referência. Diante dessa situação, conectar várias redes de maneira transparente era um dos principais objetivos do projeto ARPANET (COMER, 2005).

Prevendo a possibilidade de perda de seus *hosts*[1], roteadores[2] e *gateways*[3], o DoD definiu também que a conexão deveria ser mantida enquanto a origem e o destino permanecessem em funcionamento, mesmo com a perda de um *hardware* intermediário de sub-redes , além de ser capaz de suportar diferentes tipos de aplicações (TANENBAUM, 2003).

Hoje a arquitetura ARPANET é conhecida como Modelo de Referência TCP/IP em virtude de seus dois principais protocolos *Transfer Control Protocol* e *Internet Protocol.*

2.2. Organização do TCP/IP

Projetos de protocolos são mais complexos quando se pretende resolver os problemas como um todo. Dentre os problemas tem-se: falha de *hardware*, congestionamento de redes, demora ou perda de pacotes, danificação de dados e duplicação ou erros seqüenciais. Considerados em conjunto, a complexidade aumenta (COMER, 2005).

---

[1] Qualquer computador ou máquina ligado a uma rede.
[2] Equipamento utilizado para promover a comunicação entre duas redes.
[3] São considerados pontos de ligação entre redes, assim como os roteadores. Pode, também servir como tradutor de protocolos e separador de domínios de colisão.

Sendo assim, o projeto da maioria das redes de computadores é organizado como uma pilha de camadas (ou níveis) uma sobre as outras, de forma que cada uma resolve um determinado tipo de problema e oferece serviços para as camadas acima.

Seguindo a proposição anterior, o protocolo TCP/IP é organizado em camadas. Por ser um padrão de fato[4] não existe um modelo oficial, ou seja, foi consagrado naturalmente, sem qualquer plano formal. Surgiu através de pesquisas que levaram à pilha de protocolos atual. Com algumas modificações no modelo OSI, pôde-se propor as camadas do protocolo TCP/IP, porém as bases são diferentes para garantir distinção entre os dois modelos.

A divisão das camadas varia de acordo com o autor. Conforme Comer (2004), a pilha de protocolos TCP/IP é organizada em quatro camadas conceituais (interface de rede, inter-rede, transporte e aplicativo) as quais são construídas sobre uma quinta camada de *hardware* não pertencente ao modelo. Já de acordo com Stallings (2005), essa quinta camada faz parte da pilha de protocolos, tendo como conjunto final às camadas física, de acesso à rede, de inter-rede, de transporte e aplicação.

Este trabalho será embasado no modelo em quatro camadas explorado por Comer (2005), uma vez que a referente obra é específica para a interligação em redes TCP/IP.

Para melhor esclarecer as atividades de cada camada serão apresentadas quatro sessões, uma para cada nível do protocolo e, dentro delas, outras explorando aspectos relevantes de cada uma delas.

2.2.1. Camada de Interface de Rede

A camada de interface de rede constitui o nível mais baixo do *software* de comunicação. É responsável pela aceitação de datagrama IP e por sua transmissão através de

uma determinada rede. Uma interface de rede pode ser representada por um subsistema que utiliza seu próprio protocolo de enlace (COMER, 2005).

A arquitetura Internet TCP/IP não restringe qualquer rede interligada para formar a inter-rede (COLCHER *et al*., 2004). Modelos diferentes podem ser acoplados desde que se desenvolva uma interface que compatibilize a tecnologia a ser utilizada com o protocolo IP. O nível de interface é responsável por essa normalização no que se refere à tecnologia: o datagrama IP é recebido e encaminhado a uma rede específica através da tradução de endereços lógicos (endereços IP) para endereços físicos dos *hosts* e *gateways*.

De acordo com Stallings (2005) o emissor precisa fornecer à rede o endereço do destinatário para que a rede possa rotear os dados para o destino apropriado. Além disso, o computador de envio pode solicitar serviços com prioridade, os quais poderiam ser oferecidos pela rede.

Segundo Tanenbaum (2003) a camada em questão encontra-se como sendo a camada *host*/rede, na qual raramente é especificado o protocolo utilizado, podendo este variar de acordo com a rede ou o *host*.

2.2.2. Camada Internet

Comer (2004) explora a idéia de que a camada Internet é responsável pelo tratamento da informação entre duas máquinas. O nível de transporte solicita o envio de um pacote e fornece a identificação do endereço destino. Ao aceitar a solicitação, a camada Internet encapsula o pacote recebido em um datagrama IP, preenche o seu preâmbulo e utiliza um

[4] Padrão não especificado por uma norma, consagrado naturalmente.

algoritmo de roteamento para decidir se o pacote será entregue diretamente ou se será mandado a um *gateway*.

O nível inter-rede, denominação atribuída por Tanenbaum (2003) à camada Internet, também processa pacotes provenientes da camada de interface de rede. Nesse contexto, o algoritmo de roteamento decide se o datagrama deve seguir a hierarquia do protocolo e seguir para a camada de transporte ou se deve ser passado a diante utilizando uma interface de rede.

Um formato de pacote oficial é o protocolo IP (*Internet Protocol*), explorado na próxima sessão.

2.2.2.1. IP (*Internet Protocol*)

Segundo Colcher (2004), o protocolo IP foi projetado com o intuito de permitir a interconexão de redes computacionais, as quais utilizam tecnologia de comutação de pacotes. Sua função é transferir conjunto de dados, chamado datagrama, da origem para o destino, *hosts* identificados por endereços IP. Além disso, o protocolo em questão pode realizar fragmentação e remontagem de datagramas longos.

O serviço oferecido é sem conexão, sendo assim, os datagramas são tratados como unidades independentes sem qualquer relação com os demais. Não ocorre reconhecimento fim-a fim ou entre nós intermediários, caracterizando uma comunicação não confiável. Além disso, não se utiliza controle de erros nos dados transmitidos, apenas um *checksum* do cabeçalho para garantir a veracidade das informações nele contidas, uma vez que são requeridas pelo *gateway*.

Colcher (2004) mostra a organização dos campos de um datagrama da Internet conforme a Figura 4, considerado a unidade básica de transferência em uma interligação em redes TCP/IP. O campo *vers* especifica a versão do protocolo, ao qual o datagrama pertence. É interessante a utilização desse campo para controlar a transição entre as versões, o que pode levar meses ou até anos.

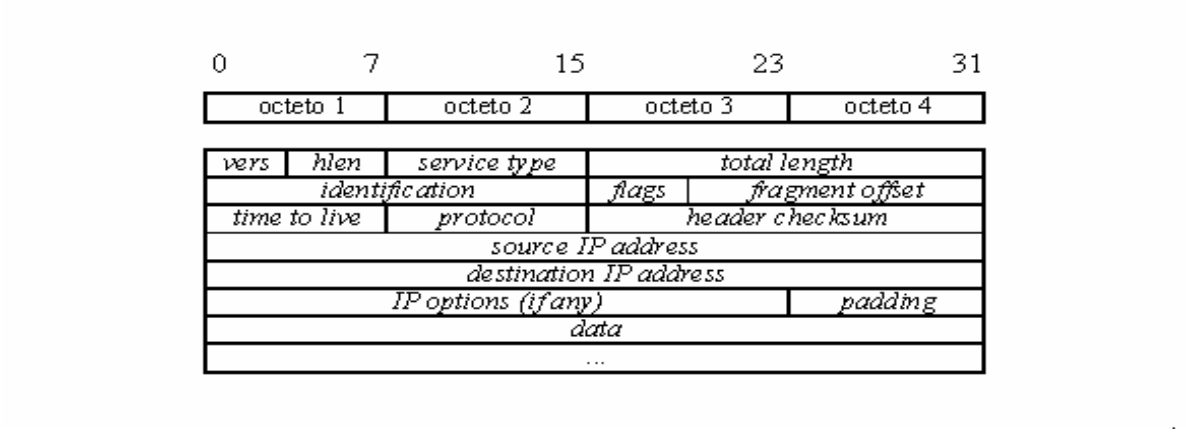

Figura 5: Representação do Datagrama IP (Colcher *et. \al.*, 2004).

Existe um campo *Hlen* de 4 *bits* para informar o tamanho do cabeçalho em palavras de 32 *bits*. É necessário esse controle, pois o tamanho do cabeçalho não é constante. O valor mínimo é 5 *bits*, enquanto que o máximo é 15, limitando o cabeçalho em 60 *bytes*. O armazenamento de parâmetros para determinar a qualidade do serviço prestado pelas redes por onde o datagrama passa está no *service type.* Este é um dos poucos campos que tiveram seu significado alterado ao longo dos anos.

O *time to live* indica o limite de tempo de transmissão dos datagramas. Na criação do datagrama é atribuído um valor a esse campo e a cada retransmissão por um *gateway* ele é decrementado. Quando chegar ao valor zero o datagrama é descartado.

O campo *protocol* indica o protocolo de transporte. O *header checksum* é utilizado pelos *gateways* para verificar a veracidade das informações contidas no cabeçalho. O *source IP address* e *destination IP address* contêm, respectivamente, o endereço do *host* origem e

destino. Por fim, o campo *IP options* é usado para informações de segurança, relatório de erros, roteamento, dentre outros serviços e o *data* carrega os dados do datagrama IP.

Para fazer referência aos *hosts* são utilizados endereços IP, números com 32 *bits,* geralmente escritos como quatro octetos, como por exemplo, *192.168.1.1.*. A primeira parte identifica uma rede na inter-rede e a segunda identifica um sistema final dentro dessa rede. O endereço IP pode não representar uma máquina individual, mas uma conexão à inter-rede (COLCHER *et al*., 2004).

Stallings (2005) acrescenta que esse protocolo não é apenas implementado em sistemas finais, mas também em roteadores. Este repassa dados de uma rede para outra em uma rota a partir da origem até o destino.

2.2.3. Camada de Transporte

Semelhante à camada de transporte do modelo de referência OSI, o nível de transporte TCP/IP tem como finalidade permitir a conversação fim a fim entre os *hosts* de origem e destino (TANENBAUM, 2003).

Além disso, essa camada pode fornecer transporte confiável para garantir a chegada sem erros e em seqüência dos dados, regular o fluxo de informações e dividir o conjunto de dados em pequenas partes para passá-las à camada seguinte, juntamente ao endereço destino, para serem transmitidos. Outros parâmetros de controle podem ser acoplados ao pacote dependendo da aplicação: código de identificação do programa emissor e receptor tais como uma soma, para haver certificação pelo aplicativo o qual irá receber o pacote.

Foram definidos dois protocolos fim-a-fim para essa camada: o primeiro é o TCP (protocolo orientado a conexões confiáveis e com entrega de um fluxo de *bytes* sem erros ao destino, será explorado na sessão 2.2.3.1. TCP (*Transmission Control Protocol*); e o segundo é o UDP (*User Datagrama Protocol*), sem conexão e não confiável dirigido a aplicações que não solicitam controle de fluxo e nem manutenção da seqüência das mensagens enviadas.

2.2.3.1. TCP (*Transmission Control Protocol*)

O TCP é um protocolo orientado à conexão projetado especificamente para oferecer um fluxo de *bytes* fim a fim em uma inter-rede não confiável. Entende-se como inter-rede um conjunto de redes distintas com características divergentes tais como topologia, largura de banda, retardos, tamanho do pacote e outras (TANEBAUM, 2003).

A formalização do protocolo foi estabelecida primeiramente na RFC 793 (ISI, 1981). Com o aparecimento de erros e inconsistências muitos requisitos mudaram em algumas áreas, o que originou a RFC 1122 (BRADEN, 1989).

Cada máquina compatível com o protocolo em questão possui uma entidade TCP (procedimento de biblioteca, processo do usuário ou parte do núcleo) responsável pelo gerenciamento dos fluxos e interfaces com a camada IP. A entidade aceita conjunto de dados proveniente de processos do usuário, e os divide em blocos menores de 64Kb no máximo para posteriormente enviá-los em datagramas IP distintos. Além disso, esse *software* (entidade TCP) realiza a restauração dos fluxos de dados originais conforme os datagramas IP chegam.

A maioria dos protocolos confiáveis utiliza a *confirmação positiva com retransmissão* como técnica fundamental para oferecer uma transferência confiável se o sistema de

comunicação disponibilizar apenas transmissão de pacotes não confiáveis (COMER, 2005). Antes de enviar o próximo bloco de dados o transmissor necessita receber uma confirmação do receptor chamada ACK (*acknowledged*). Caso esse reconhecimento não chegue, o segmento é retransmitido. Isso é possível pelo fato de a entidade TCP guardar uma cópia deste segmento em uma fila de retransmissão, da qual apenas sairá quando chegar a mensagem de reconhecimento corretamente ou quando um temporizador, ativado na entrada do seguimento, expirar.

Outro serviço previsto pelo protocolo TCP é o controle de fluxo. Colcher (2004) explica que este controle é baseado no envio da capacidade de recepção do sistema final receptor, ou seja, fornece o tamanho da *janela de recepção*. Com esse parâmetro o transmissor calcula qual o número de segmentos propício para envio.

O TCP utiliza o conceito de *porta* para permitir o acesso aos seus serviços por vários processos em um único *host.* Cada usuário (aplicações) atendido pelo protocolo em determinado momento é identificado por uma porta diferente. Como os números das portas podem se repetir ao longo da inter-rede, concatenam-se este número ao endereço IP para formar um *socket* (COLCHER *et. al*., 2004).

2.2.3.2 Controle de Congestionamento

Outra característica interessante e que merece atenção é o controle de congestionamento oferecido por alguns protocolos da camada de transporte, tais como TCP e HSTCP (*Highspeed* TCP).

Conforme Comer (1999) congestionamentos originam retransmissão de segmentos, e assim geram colapso de congestionamento. Para se evitar esse quadro, o TCP especifica que uma estratégia deve ser utilizada para diminuir a quantidades de retransmissões na ocorrência de atrasos ou perda de pacotes.

Já Tanenbaum (2003) diz: "quando a carga oferecida a qualquer rede é maior que sua capacidade, acontece um congestionamento". Em sua abordagem ele menciona o controle de congestionamento realizado pela camada de rede, porém ressalta que o trabalho mais árduo é feito pelo TCP na camada de transporte, pois a diminuição da taxa de transmissão é a solução para o congestionamento. Isso é proferido através da manipulação dinâmica do tamanho da janela.

São mantidas duas janelas pelo transmissor: a fornecida pelo receptor e a de congestionamento. As janelas definem a quantidade de seguimentos que podem ser enviados sem ter recebido confirmação (ACK). O valor mínimo entre as duas janelas será o valor permitido para transmitir (TANENBAUM, 2003).

Ao estabelecer uma conexão, o transmissor define a janela de congestionamento com o valor máximo de segmento (MSS) em uso na conexão. Logo após, ele envia um MSS. Caso seja confirmado antes da ocorrência de um *timeout*, o transmissor inclui mais um seguimento à janela de congestionamento, de forma que ela tenha a capacidade de dois seguimentos e assim enviar esse número. Na prática, cada transmissão bem sucedida duplica a janela de congestionamento (TANENBAUM, 2003).

A janela pára de crescer quando ocorre um *timeout* ou quando a janela do receptor é alcançada. Caso a janela do receptor possua espaço, mas a janela de congestionamento não indica que esse espaço deva ser preenchido, o número de *bytes* transmitidos será aquele indicado pela janela de congestionamento. Esse algoritmo é conhecido como partida/início lenta(o) (*slow start)* (COMER, 2004).

De acordo com Comer (1998) a cada transmissão em uma conexão recente é utilizado o início lento para o congestionamento não se agravar com essas novas conexões.

A Figura 5 ilustra o algoritmo de congestionamento da Internet, o qual possui mais um parâmetro para representar a quantidade máxima de seguimentos a serem enviados, ou seja, um limiar (*threshold*). Percebe-se, na Figura 6, o crescimento exponencial da janela até alcançar o valor do limiar, em seguida o crescimento é linear.

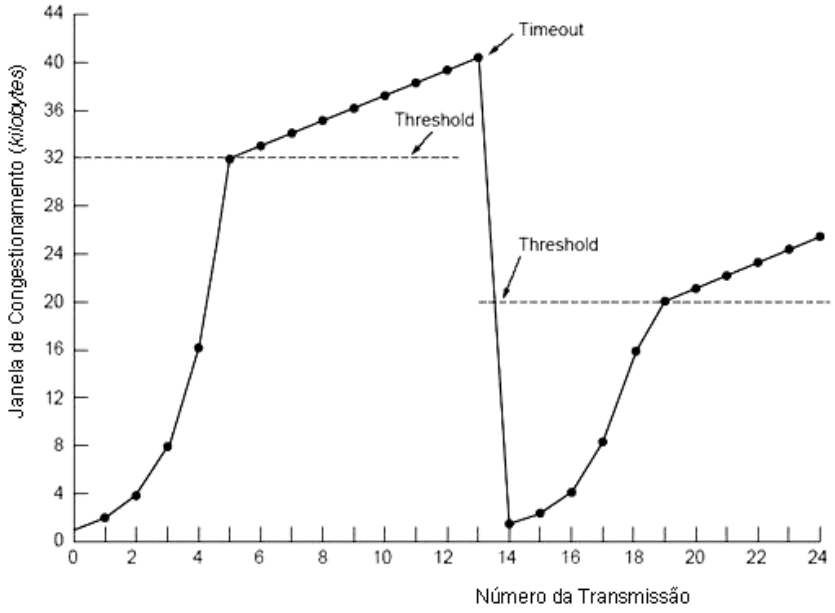

Figura 6: Algoritmo de congestionamento da Internet. (TANENBAUM, 2003)

Conforme Rezende (2005), o protocolo TCP mesmo sendo muito utilizado em transmissões pela Internet, seu mecanismo de controle de congestionamento limita o tamanho da janela de congestionamento a qual pode ser alcançada pelas redes de alta velocidade.

Vários protocolos de transportes direcionados para redes *Gigabits* estão surgindo tais como: XCP (*Explicit Congestion Control Protocol*) (FALK *et. al.,* 2003), HSTCP (*HighSpeed TCP*) (FLOYD, 2003), STCP (*Scalable* TCP) e o FAST TCP (*Fast Active-queue-management Scalable TCP*) (JIN *et. al.,* 2003). O protocolo XCP exige mudanças nos roteadores. O HSTCP, STCP e o FAST TCP são implementados nos sistemas finais, porém esse último é

baseado no TCP Vegas, uma variante pouco utilizada na Internet. O STCP é baseado no HSTCP (REZENDE, 2005). Devido às características citadas acima, o HSTCP foi escolhido para estudos no presente trabalho.

O protocolo HSTCP foi proposto pela RFC 3649 (2003) como uma modificação do mecanismo de controle de congestionamento do TCP para ser utilizado, juntamente com conexões TCP, em redes com grandes janelas de congestionamento. Para especificar uma função de resposta modificada foram usados três parâmetros: *Low_Window, High_Window* e *High_P*. Para garantir a compatibilidade do TCP, o *HighSpeep* utiliza a mesma função de resposta do TCP quando a janela de congestionamento corrente está, em sua maioria, para *Low_Window* e faz uso do a função de resposta do *HighSpeep* quando a janela de congestionamento é maior que *Low_Window*.

2.2.4. Camada de Aplicativos

Diferentemente do modelo de referência OSI, a arquitetura TCP/IP não possui as camadas de sessão e de apresentação. Esse fato se deve a não utilização freqüente dessas camadas no modelo OSI (TANENBAUM, 2003). Sendo assim, logo acima do nível de transporte, encontra-se o de aplicação.

Stallings (2005) atribui à camada de aplicação a lógica necessária para suportar diversas aplicações do usuário. Cada tipo necessita de um módulo próprio peculiar a essa aplicação. Não existe um padrão estrutural como no modelo OSI (Figura 7).

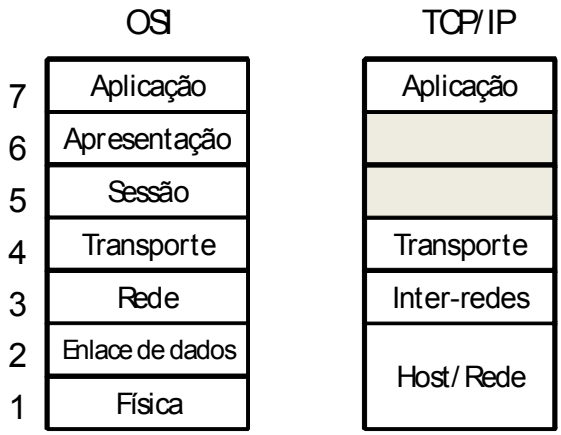

Figura 7: O modelo de referência TCP/IP em relação ao modelo OSI (TANENBAUM, 2003).

O nível de abstração da camada quatro (aplicação) é o mais alto da arquitetura como ilustra a Figura 8. Dentre os protocolos dessa camada estão: protocolo de terminal virtual (TELNET), o para transferência de arquivos (FTP) e o de correio eletrônico (SMTP). O primeiro permite a conexão de um usuário a uma máquina fisicamente separada, o segundo fornece o movimento eficiente de dados de uma máquina para outra e finalmente o terceiro é utilizado na troca de mensagens eletrônicas (TANENBAUM, 2003).

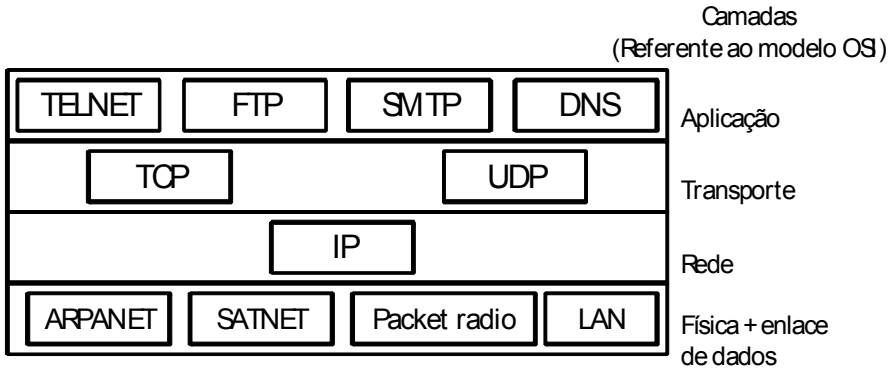

Figura 8: Protocolos utilizados em cada camada da arquitetura TCP/IP.

Com o decorrer dos anos, outros protocolos foram incluídos, como por exemplo, o DNS que mapeia os nomes de *hosts* para os endereços de rede correspondentes e o HTTP

utilizado para buscar páginas na *Word Wide Web*. A disposição dos protocolos descritos é mostrada na Figura 8.

2.3 Futuro do TCP/IP

Durante uma década a tecnologia TCP/IP funcionou bem. Entretanto, novos parâmetros surgiram e ameaçam mudar alguns de seus aspectos. Comer (2004) lista quatro categorias que impulsionam a evolução desse protocolo.

Primeiro estão as novas tecnologias de computadores e comunicação, com o emprego de sistemas computacionais de alto desempenho para funcionarem como roteadores e *hosts*, além do aparecimento de estudos sobre transmissão via satélite e ATM. Segundo, os novos aplicativos que exigem uma demanda de recursos não oferecidos pelos protocolos atuais tais como áudio e vídeo em tempo real. Terceiro, o aumento exacerbado do tráfego. E por fim, o quarto, representado pelo surgimento de novas jurisdições, as quais levam a novas políticas administrativas.

A princípio o tamanho de 32 *bits* para o endereçamento IP parecia suficiente, no entanto, os projetistas não previram o grande aumento das redes em organizações de médio porte e a associação de empresas grandes com redes remotas. Assim, a questão do endereçamento passa a ser a grande motivação para a criação de uma nova versão do protocolo.

O protocolo IP*v*6 mantém muito das diversas características fundamentais ao sucesso do IP*v*4. Comer (2004) agrupa as principais mudanças em cinco categorias: Endereços Maiores (mudança mais visível), Formato Flexível de Cabeçalho com um novo e arrojado

formato de datagrama (Figura 9), Opções Aprimoradas (permite a inclusão de informações de controle adicionais assim como na versão anterior), Suporte para Alocação de Recursos e por fim, Provisão para Extensão de Protocolos.

Formalmente foi decidido o número 6 para a nova versão do protocolo IP. Para diferenciá-la da atual versão (IP*v*4), a próxima será chamada de IP*v*6 (COMER, 2005). No passado, o termo IP*ng* (IP *Next Generation*) foi amplamente utilizado para referenciar todas as propostas e discussões sobre a versão posterior do IP, enquanto que o IP*v*6 era aplicado para fazer referência à proposta surgida da IETF (*Internet Engineering Task Force*). Atualmente os dois termos são encontrados na literatura.

Conforme Tanenbaum (2003), o IP*v*6 não é compatível com a versão quatro, mas com todos outros protocolos auxiliares da Internet tais como TCP, UDP, DNS e outros, apesar de precisar de certas modificações em determinados momentos. A especificação desse aperfeiçoamento do protocolo IP está detalhado na RFC 2460 (NWG, 1998).

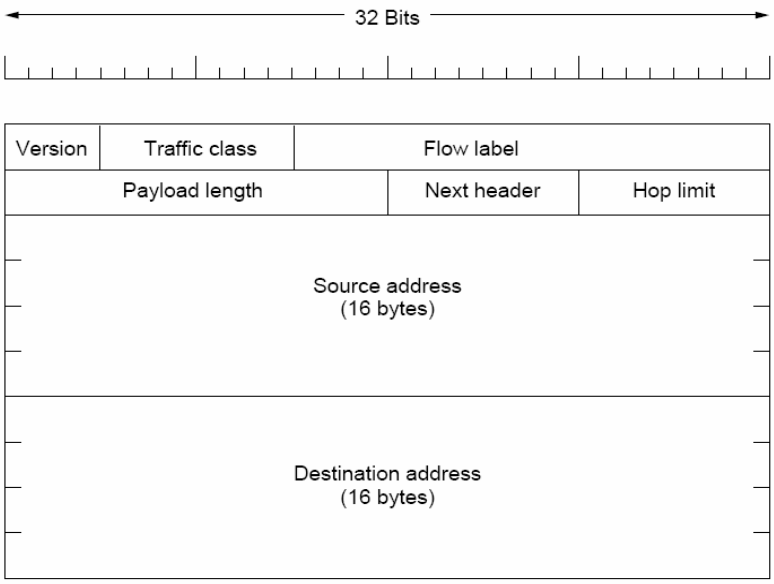

Figura 9: Cabeçalho fixo do IP*v*6 (TANENBAUM, 2003).

2.4 Considerações Finais

Com a evolução dos sistemas computacionais juntamente aos avanços na área de redes de computadores, novas funcionalidades tiveram que ser atribuídas aos protocolos projetados a princípio. Esse fato é claramente observado na transição do IP comumente utilizado para a recente versão (IPv6).

Outro fato relevante é o aumento desordenado de *hosts* acoplados à rede mundial de computadores, exigindo novas tecnologias para suprir essa demanda. Surge, então, um novo conceito para tornar a criação de novos protótipos tecnológicos mais viáveis e seguros. Essa nova vertente é a simulação.

A parte prática dessa monografia será fundada na simulação de redes com topologia estrela utilizando o simulador *Network Simulator*, a fim de observar e listar as vantagens desse tipo de ferramenta, bem como o desempenho do estudo de caso realizado.

3. SIMULAÇÃO

De acordo com o significado do dicionário, *simular* significa imitar, representar com semelhança. Sendo assim, simular um modelo computacional, é representar, com maior semelhança possível, uma situação desejada.

A simulação de redes tem-se mostrado uma solução para a realização de testes vislumbrando a redução do alto custo da montagem de laboratórios específicos para esse fim.

Neste capitulo serão apresentadas vantagens e características da utilização de simulações de redes. Em simulação de redes de computadores serão mostradas algumas técnicas de simulação e trabalhos os quais fazem uso de ferramentas de simulação. Em seguida, serão apresentados alguns dos simuladores mais utilizados, sendo a seção 3.3.4. Network Simulator (NS-2)dedicada ao *Network Simulator*.

3.1. Simulação de redes de computadores

A análise de desempenho e a modelagem de redes de computadores são tarefas complexas e que exigem certo trabalho. De acordo com Dias (*et. al.* 2003) três soluções geralmente podem ser utilizadas para auxiliar a execução dessas tarefas: experimentação com redes reais, métodos estatísticos e simulações.

A realização de experimentos em redes reais oferece vantagem em relação aos resultados obtidos, as quais são mais consistentes por serem reais. Por outro lado, existem algumas desvantagens nesse tipo de método tais como: alto custo de implementação, tempo

gasto para a construção da rede, preparação do ambiente de validação, desenvolvimento de técnicas para geração e coleta de tráfego de dados.

A utilização de métodos analíticos baseia-se na substituição da rede por modelos matemáticos possíveis de serem resolvidos analiticamente. É um método eficiente, porém em determinadas situações, tais como no contexto de redes de alta velocidade, apresenta-se pouco eficiente por fazer uso de um número elevado de variáveis e, assim, na ocorrência de simplificações a conclusão obtida tornam-se limitadas.

Considerando os aspectos mencionados anteriormente, foi escolhida a técnica de simulação de redes para a realização do presente trabalho, uma vez que possui maior flexibilidade em relação aos modelos apresentados. É baseada na construção do modelo de maneira eficiente com base nas respostas esperadas pela simulação. Outras vantagens observadas na utilização da técnica são: facilidade na realização de testes, nível de detalhamento mais apurado na modelagem, comparação entre diferentes mecanismos de controle de tráfego, gerenciamento de recursos, dentre outras (DIAS *et. al.,* 2003).

A simulação de redes é importante tanto no planejamento quanto no desenvolvimento de uma rede, com o intuito de avaliar diversos pontos básicos e também evitar a ocorrência de erros.

Existe na literatura a especificação de diversos simuladores de redes. Cada qual possui características que os diferenciam ou os tornam melhores. Na seção 3.2. serão apresentados alguns desses simuladores.

Para comprovar a crescente utilização do simulador NS-2 em pesquisas, serão apresentadas, a seguir, descrições rápidas sobre trabalhos presentes na literatura aberta referentes à área de simulação de redes de computadores.

Trabalhos utilizados como revisão bibliográfica para a presente pesquisa têm sido realizados desde 1992 como: NetSim: A *Network Performasce Simulator* e NetSin *User's*

Manual (1994), desenvolvidos pelo matemático e cientista da computação Lewis Barnett do Departamento de Matemática e Ciência da Computação da Universidade de Richmond, Virginia. São materiais que abordam os conceitos e funcionalidades da ferramenta *Network Simulator*. Os documentos mais atuais relacionados ao NS estão presentes em (FALL, 2006).

Outro indicativo da eficácia do simulador em questão é a publicação, pela revista IEEE *Computers*, de (BLESLAU *et al*., 2000) referente aos avanços na área de simulação de redes. Aborda características tanto estruturais (arquitetura do *software*), quanto à qualidade dos resultados, especificando porque essa qualidade é obtida.

No Brasil existe um gama de projetos envolvendo o uso da ferramenta *Network Simulator*: O Algoritmo SQM-Response para Controle de Congestionamento do Protocolo TCP, projeto que propõe a criação de um algoritmo (SQM-*Response*) para notificar explicitamente o congestionamento em redes de computadores com a finalidade de melhorar o desempenho do TCP (ANDRADE *et. al.,* 2003) . Nesse trabalho o NS foi utilizado para avaliação de desempenho do algoritmo; Estudo do Comportamento do TCP/IP em Enlaces Via Satélite, trabalho de graduação em Engenharia de Redes que vincula o uso as tendências do uso da Internet (através do entendimento da arquitetura de protocolos TCP/IP), com a utilização de tecnologias de transmissão via satélite (OLIVEIRA, 2002).

Os comentários e levantamentos bibliográficos apresentados nesta seção, apenas reforçam a real e atual necessidade em se continuar efetuando pesquisas na área de redes de computadores, mais especificamente fazendo uso da simulação para alcançar os objetivos almejados.

3.2. Simuladores mais utilizados

Diversos simuladores surgiram à medida que a complexidade de implementação de redes de computadores aumentou. É possível encontrar simuladores proprietários ou com código aberto. A seguir são apresentados alguns deles.

3.2.1. OpNet

O OpNet é um simulador de redes, o qual pode ser utilizado na análise de desempenho de redes efetivamente implementadas ou em redes criadas no próprio *software*. Possui recursos para captura de tráfego em uma rede com variação no número de usuários que acessem serviços diferenciados (OpNet, 2006).

É uma ferramenta utilizada para especificar, simular e analisar o desempenho da rede, além de permitir a especificação de um número elevado de componentes de comunicação via satélite, redes *wireless*, redes locais, dentre outros.

Uma de suas características marcantes e que destoa do NS, é o fato de ser uma ferramenta comercial e com custo elevado. Esse fator o torna menos propício para a realização deste trabalho.

As simulações realizadas no OpNet podem ser divididas em 5 etapas como ilustra a Figura 10.

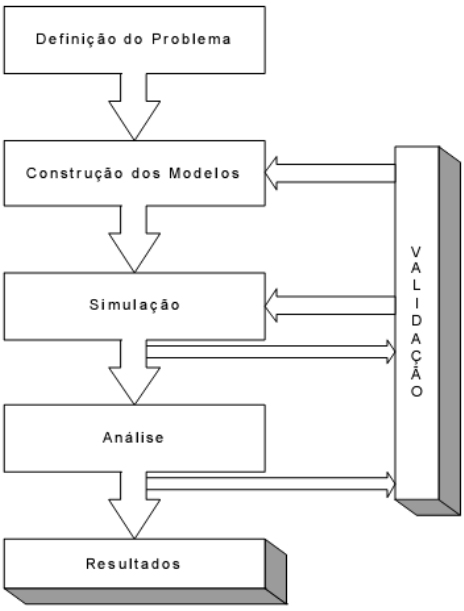

Figura 10: Etapas da modelagem e simulação com OpNet (DIAS *et.al.,* 2003).

3.2.2. GloMoSim

O GloMoSim (*Global Mobile Information Systems Simulation Library*) é um simulador de redes sem fio baseado em uma biblioteca de funções que, quando compilada e utilizada, permite a simulação de forma muito escalável, aproveitando assim a capacidade de simulação de eventos paralelos fornecido pela linguagem PARSEC (*PARallel Simulation Environment for Complex systems*), o que permite o processamento paralelo (GLOMOSIM 2003). Para executar a biblioteca de funções do GloMoSim é necessário um compilador C como o GCC (GNU C *Compiler*) e uma versão do compilador PARSEC o PCC (UCLA, 2001), que atualmente já vem incluída no pacote de distribuição do GloMoSim.

O GloMoSim implementa um conjunto de protocolos de rede para comunicação sem fio, que estão organizados em uma arquitetura em camadas. Novos protocolos e módulos

implementados em PARSEC podem ser facilmente adicionados à biblioteca do GloMoSim para compor diferentes simulações. O núcleo do PARSEC implementa uma máquina de simulação de alto desempenho que permite ao GloMoSim fazer a simulação de redes de larga escala com total transparência de simulação, tanto para o programador do protocolo quanto para o usuário do simulador.

Entretanto o GloMoSim é um simulador que tem o foco específico para simulação de redes de computação móvel de larga escala. Atualmente está em desenvolvimento, mas o GloMoSim é a base de um produto comercial denominado QualNet (QualNet, 2006), que ao contrário do GloMoSim é pago (GLOMOSIM 2003).

3.2.3. Boson NetSim

O Boson é um simulador com vários anos de utilização. Possui uma rede global de escritores e de instrutores, os quais são profissionais capacitados para oferecer três linhas de produto certificação da Cisco: *Software* de ExSim, *software* de NetSim, e treinamento em sala de aula.

Realiza a emulação de tabelas de chaveamento de pontes como também tabelas de protocolo de roteamento para permitir ao usuário ir além dos exemplos do simulador. Com o Boson NetSim é possível projetar e configurar uma rede com 40 modelos diferentes de roteador.

Este simulador de rede é muito versátil e realístico. Outros produtos simulam *scripts* para um usuário experiente, sem simular o que está acontecendo realmente dentro da rede. A tecnologia virtual de pacotes do Boson cria pacotes individuais os quais são distribuídos e

comutados através da rede simulada, o que permite ao Boson NetSim construir uma tabela de roteamento virtual apropriada e simular o trabalho em rede real.

O Boson oferece dois produtos focalizados na certificação Cisco: NetSim para CCNA e NetSim para CCNP. Cada um suporta as tecnologias e as habilidades que um usuário necessitará para as respectivas certificações. São incluídos, ainda, menus detalhados do laboratório de lições e dos responsáveis pelos protocolos de roteamento, dispositivos de Cisco, *Switching,* projeto topológico, entre outros.

3.3.4. Network Simulator (NS-2)

A Ferramenta *Network Simulator* (NS-2) foi criada em 1989 como uma variante do *Real Network Simulator* (FALL, 2006), desde então auxilia na melhor disposição dos equipamentos de uma rede, além de fornecer informações sobre o fluxo de dados e gerar gráficos demonstrativos.

É um simulador de eventos discreto e possui a grande vantagem de ser gratuito e, assim, possibilitar sua manipulação de acordo com a necessidade do usuário. Além disso, oferece suporte à simulação de um número grande de tecnologias com ou sem fio (COUTINHO, 2003).

3.2.4.1. Características

O NS-2 é um simulador de redes dirigido a eventos desenvolvido na UC Berkeley (*Berkeley University of California*) capaz de simular variedades de redes IP. Nele é possível implementar protocolos de redes tais como TCP e UDP, comportamento de fontes de tráfego (FTP, Telnet, Web, CBR e VBR), mecanismos de manipulação de filas em roteadores (*Drop Tail*, Red e CBQ), algoritmos de roteamento como Dijkstra entre outros. Também implementa *multicasting* e alguns protocolos da camada MAC para simulações de LANS.

Atualmente o projeto do NS faz parte do projeto VINT (VINT, 1997), o qual desenvolve ferramentas para análise, demonstração e conversão de topologias de redes para o formato do NS.

A Figura 11 mostra, de maneira simplificada, que o NS é um interpretador de *scripts* Tcl orientado a objeto (OTcl), o qual possui um esquema de simulação de eventos, biblioteca de objetos componentes da redes e bibliotecas para configuração da rede. Para configurar e executar uma simulação, o usuário deve escrever um *script* OTcl que inicie o escalador de eventos, estabelecer a topologia de rede e fazer chamada às fontes de tráfego quando começar e terminar a transmissão de pacotes.

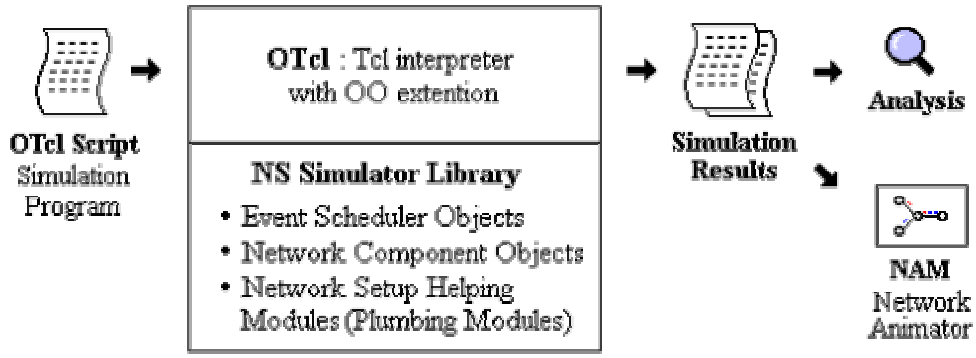

Figura 11: Visualização simplificada do NS-2.

A hierarquia de classes do NS é ilustrado na Figura 12. A raiz da hierarquia é a classe *TclObject*, a qual é a superclasse de toda biblioteca de objetos Otcl. Posteriormente encontra-se a classe *NsObject*, superclasse de todos objetos referentes ao componentes básicos de uma rede. Esses componentes básicos são divididos em duas classes *Connector* e *Classifier*, de acordo com a quantidade de trajetos possíveis para saída de dados.

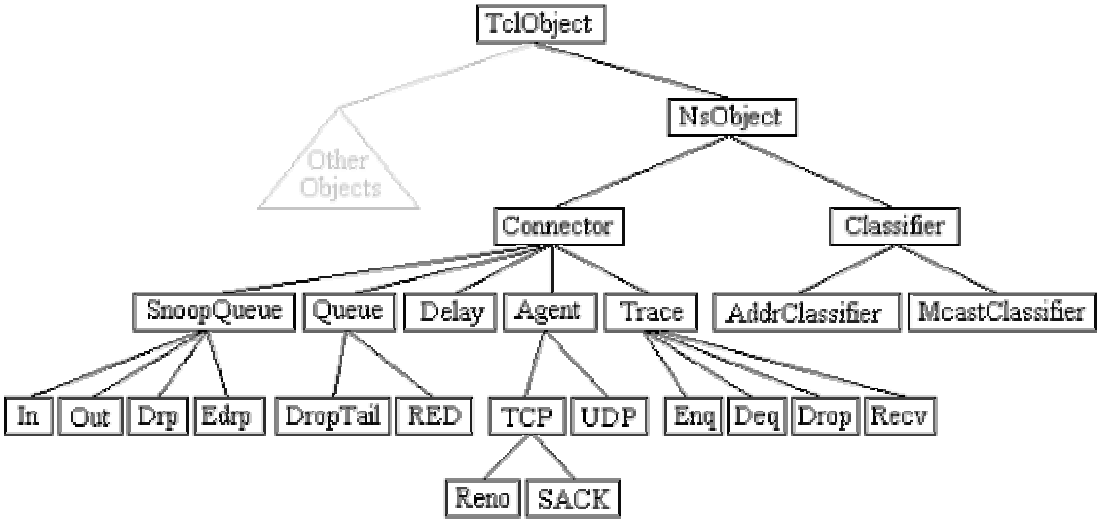

Figura 12: Hierarquia parcial de classes no NS (CHUNG *et .al.*, 2003).

3.2.4.2. Programação no NS-2

A programação no NS-2 pode ser realizada em duas linguagens:

- C++: linguagem compilada e tradicional usada para manipulação de *bytes*, pacotes e para aplicações com manipulação de um grande conjunto de dados;
- Otcl (*Object-oriented Tool Command Language*): linguagem interpretada e orientada a objeto para confecções dos *scripts* de simulação e eventuais ajustes nos parâmetros do mesmo

A justificativa para a necessidade de duas linguagens diferentes é a presença de dois tipos diferentes de tarefas a serem realizadas. Uma é a implementação de simulações detalhadas de protocolos, isso exige uma robustez maior, portanto utiliza-se a linguagem C++. Por outro lado, uma grande parte das pesquisas de rede consiste em variar, ligeiramente, parâmetros e configurações, ou mesmo explorar rapidamente um número de cenários. Nestes casos o mais importante é o tempo de interação, portanto utiliza-se uma linguagem interpretada (Otcl) para propiciar a realização de mudanças nos modelos simulados mais rapidamente.

A junção entre a linguagem C++ e Otcl é realizadz através do tclcl, conjunto de módulos específicos que acompanha o NS-2 (Figura 13). Através de tclcl, uma classe escrita em C++ pode ser instanciada usando-se código Otcl, e qualquer parâmetro modificado nesse código Otcl será refletido no objeto C++ instanciado (CHUNG *et .al.*, 2003). O total dos elementos juntos forma o NS.

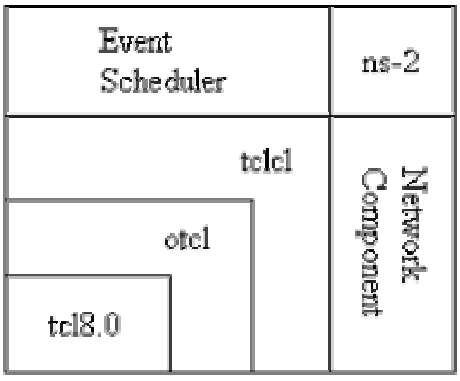

Figura 13: Arquitetura geral do NS (CHUNG et .al., 2003).

A ferramenta em questão possui diversas funcionalidades para a confecção dos scripts, porém no presente trabalho serão descritas (na seção 4.1 Materiais e Métodos) apenas aquelas efetivamente utilizadas nas simulações proferidas.

3.2.4.3. Interface gráfica

As simulações realizadas com NS podem ser visualizadas através de uma ferramenta gráfica chamada NAM (*Network Animator*). É uma ferramenta baseada em Tcl/TK para demonstrar os traços da simulação e as características reais do pacote. A teoria por trás do projeto era criar um animador capaz de ler séries de dados grandes e ser extensível o bastante de modo que pudessem ser usadas para diferentes visualizações de rede (FALL, 2006).

A primeira providência a ser tomada para realizar uma animação é gerar um arquivo de traços (*trace file*), o qual contêm informações sobre a topologia, os nós, as ligações feitas entre os nós (*links*), além dos traços (características) dos pacotes. Quando o arquivo de traços é gerado a animação está pronta para ser executada no NAM. Após a inicialização da ferramenta, a mesma providencia a leitura do arquivo de traços para formar a topologia, bem como os nós e ligações que a compõe (FALL, 2006).

A Figura 14 ilustra como criar um arquivo de traços. *Ntrfd* é a variável que recebe a localização do arquivo, cujo nome está contido em *opt(namtr)* previamente estabelecido. A segunda linha representa o armazenamento efetivo dos dados da simulação no arquivo anteriormente estabelecido.

```
set ntrfd [open $opt(namtr) w]
$ns namtrace-all $ntrfd
```

Figura 14: Código pra gerar um arquivo de traços.

Para um melhor controle por parte do usuário, o NAM proporciona uma interface com botões no estilo VCR (*play*, *fast forward*, *rewind*, *stop*). Assim pode-se escolher o momento

da simulação desejável para visualização, bem como, redefinir a apresentação do desenho da topologia (FALL, 2006).

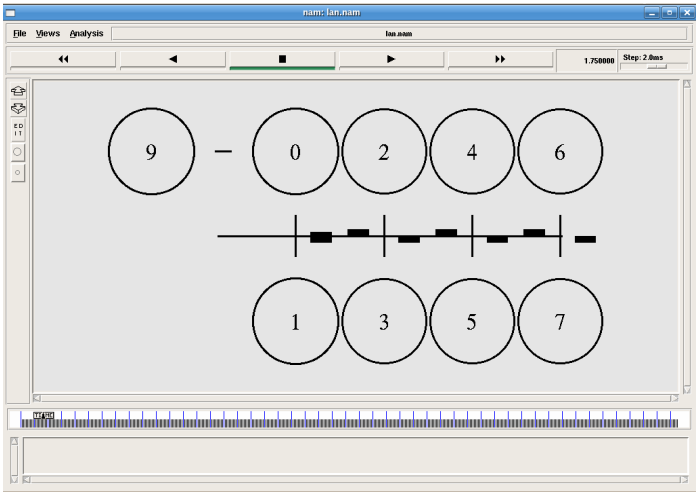

Figura 15: *Network Animator* – topologia utilizada.

A Figura 14 ilustra a visualização que o NAM oferece. Essa animação foi adquirida na simulação de uma LAN com oito nós e um agente externo, o qual injeta pacotes na rede. Esse *script* será mais bem explicado nas seções posteriores.

3.2.5. Considerações finais

O presente Capítulo abordou aspectos relevantes sobre simulação de redes de computadores, mostrando sua eficiência e versatilidade. Outro ponto explorado foi a ferramenta *Network Simulator* muito utilizada no meio acadêmico.

Sendo assim, o próximo Capítulo é dedicado à explicação dos *scripts* de simulação bem como dos resultados obtidos. Além disso, será descrito todo o ambiente utilizado para a realização das simulações.

4. METODOLOGIA E RESULTADOS

Este Capítulo apresenta os mecanismos e materiais utilizados para a realização do trabalho. Serão mencionados os ambientes de simulação, bem como as respectivas avaliações dos resultados.

4.1 Materiais e Métodos

Os resultados foram obtidos com o auxílio de um microcomputador com processador AMD Duron 1.6 GHz com 256 Mb de memória RAM e sistema operacional Fedora *core realease* 5, *Red Hat* 4.1.0-3.

Para adquirir o *Network Simulator* basta apenas realizar o *download* do arquivo na Internet em (FALL, 2006). São oferecidos dois modelos de instalação, um contendo todos os módulos do NS (pacote *allinone*) e o outro deixando a critério do usuário quais módulos ele quer instalar, esse último é indicado a usuários mais experientes. O simulador encontra-se, atualmente na versão 2.30.

Foi utilizado o pacote *allinone.tar.gz* para a instalação, uma vez que contém todos os requisitos necessários para a realização das simulações e, possui menor dificuldade em seu manuseio.

Primeiramente o arquivo citado acima foi descompactado por meio da linha digitada no console do sistema operacional: *tar -vzxf allinone.tar.gz*. A descompactação pode ser realizada em qualquer diretório. Feito isso, basta digitar a linha *./install* na pasta *ns-allinone-*

2.29 para realizar a instalação. Depois de concluída a instalação é exibida uma mensagem com as alterações necessárias nos arquivos do *Linux*. Em seguida é aconselhável a execução do comando *./validate*, o qual deve ser executado na pasta *ns-allinone-2.29/ns-2.29.*

A confecção dos *scripts* pode ser realizada em qualquer editor de textos, nesse caso foi utilizado gedit 2.14.0, o qual oferece suporte para diversas linguagens incluindo Tcl.

4.2. Estudo de caso

A proposta do trabalho é avaliar o desempenho de uma rede TCP/IP com topologia estrela quando acrescentado um controle de congestionamento. Dois protocolos foram utilizados: TCP Reno[5] e HSTCP. A parametrização das simulações foi feita segundo os dois modelos, ou seja, com larguras de banda altas e baixas. Isso proporciona uma comparação mais pertinente entre os dois protocolos.

4.2.1. Parametrização

Dois *scripts* foram criados para a obtenção dos resultados. Os modelos (Apêndice A) simulados utilizaram o componente *LanNode*. Esse componente é criado por meio do

[5] A primeira versão do controle de congestionamento para o TCP (chamado de TCP Tahoe) usava apenas os algoritmos *Slow-Start* e *Congestion Avoidance*. O TCP Reno incorporou os algoritmos *Fast Retransmit* e *Fast Recovery*, que produzem um melhor desempenho por não reduzirem tão drasticamente a o valor da janela de congestionamento em caso de congestionamentos leves (ANDRADE et. *al.,* 2003).

comando *make-lan* ou *newLan* e possui todos os objetos compartilhados em uma LAN: *Channel* (transição de pacotes na camada física) , *Classifier/Mac* (camada de acesso ao meio) *e LanRouter* (utilizado para encontrar o próximo salto do pacote). Deste modo para cada nó

$ns_ make-lan <nodelist> <bw> <delay> <LL> <ifq> <MAC> <channel> <phy>

na LAN é criado um objeto *LanIface*, o qual contém todos os outros objetos necessários para a interligação entre dois nós: *Queue*, protocolos da camada de enlace (*LL*), MAC, entre outros. A seguir é mostrada a criação do *LanNode*:

Onde:

- *nodelist* é a lista de nós pertencentes à LAN;
- *bw* é a largura de banda na LAN;
- *delay* é o atraso, *LL* camada de enlace;
- *ifq* representa o tipo de política de fila, *MAC* camada de acesso ao meio;
- *channel* equivale ao tipo de transmissão (com fio, sem fio ou satélite); e
- *phy* simula a camada física.

Com exceção do *nodelist*, todos os outros parâmetros são opcionais.

Um objeto *LanRouter* é criado quando um novo *LanNode* é inicializado. Em todos os nós da LAN o objeto LL tem um ponteiro para o *LanRouter*, dessa forma esse nó é capaz de encontrar o próximo salto para o pacote que se encontra na rede. A Figura 15 ilustra como o roteamento é visto internamente no NS. A imagem à esquerda representa a configuração vista no NAM, e a da direita como a LAN é vista para o roteamento.

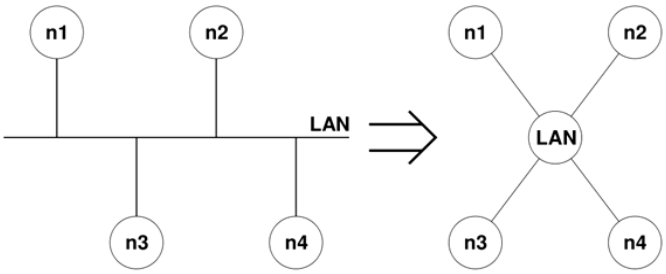

Figura 16: Visão no NAN (esquerda) e visão para roteamento (direita) (FALL, 2006).

Tendo em vista as características citadas, a parametrização das simulações foi realizada de acordo com a Tabela 1.

Tabela 1: Parâmetros para construção da LAN.

Parâmetro	***Descrição***	***Valor***
nodelist	Lista de nós pertencentes à Lan.	Oito nós: N0 à N7
Bw	Largura de banda entre as conexões da rede local.	1000Mb e 500Mb
Delay	Possíveis atrasos na rede.	2ms
LL	Responsável por simular os protocolos da camada de enlace.	LL
Ifq	Política de fila utilizada pelo roteador.	Queue/DropTail
MAC	Simula os protocolos da camada de acesso ao meio.	Mac/802_3
channel	Simula a transmissão real do pacote na camada física.	Channel
Phy	Simula o compartilhamento do meio e suporta os mecanismos de acesso ao meio do lado emissor.	-

Para criar a topologia foi construída a função *create-topology* apresentada na Figura 16.

```
proc create-topology {} {
        global ns opt node0 node1 BDP
        global lan node source node0 node1

        set num $opt(node)
        for {set i 0} {$i < $num} {incr i} {
                set node($i) [$ns node]
                lappend nodelist $node($i)
        }

        set lan [$ns newLan $nodelist $opt(bw) $opt(delay) \
                        -llType $opt(ll) -ifqType $opt(ifq) \
                        -macType $opt(mac) -chanType $opt(chan)]

        set node0 [$ns node]
        $ns duplex-link $node0 $node(0) 3000Mb 3ms DropTail
        $ns duplex-link-op $node0 $node(0) orient right
        $ns queue-limit $node0 $node(0) $BDP

#
}
```

Figura 17: Criação da topologia.

No retângulo azul aparece a criação dos nós constituintes da LAN. O retângulo em vermelho representa a criação da topologia por meio da função *make-lan* mencionada anteriormente. Nesse ponto são inseridos os parâmetros apresentados na tabela anterior. No retângulo em verde é mostrado um nó externo à LAN responsável por injetar pacotes na rede. Nota-se que sem esse nó externo, a visualização na ferramenta NAM fica desconfigurada (Figura 17), porém não foi notada qualquer alteração funcional.

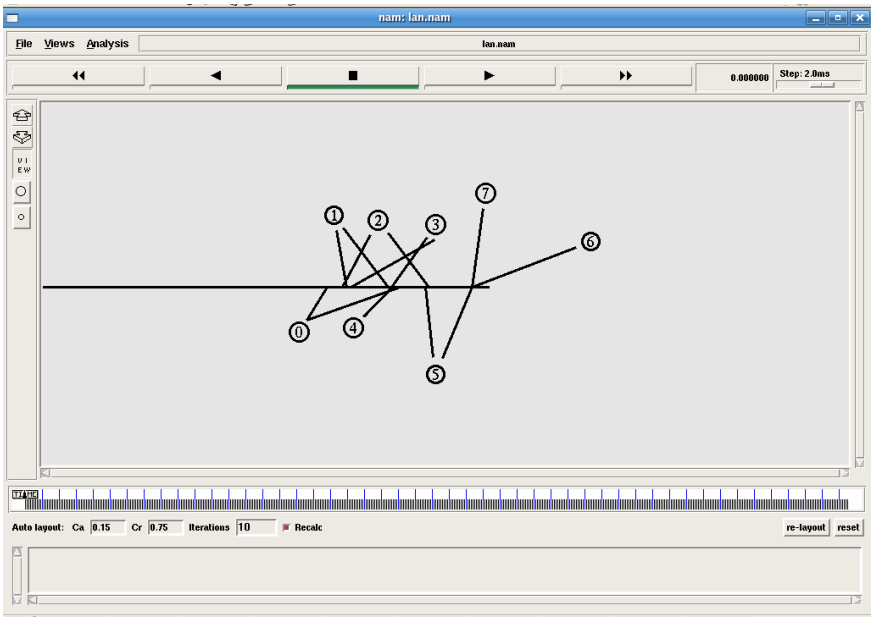

Figura 18: Visualização da LAN sem agente externo.

Criada a topologia, basta agora definir os agentes atuadores da rede. A Figura 18 ilustra a criação de um agente *TCP/Reno* (tcp0) que será anexado ao nó 0 (nó externo à LAN).

Em seguida é criado um agente *Sink* (sink) para receber os pacotes enviados pelo agente TCP e mandar a mensagem de reconhecimento (ACK). A seta azul aponta para o comando responsável pela realização da conexão entre os dois agentes, enquanto que a seta vermelha aponta para o comando o qual define o tipo da aplicação que trafegará na rede.

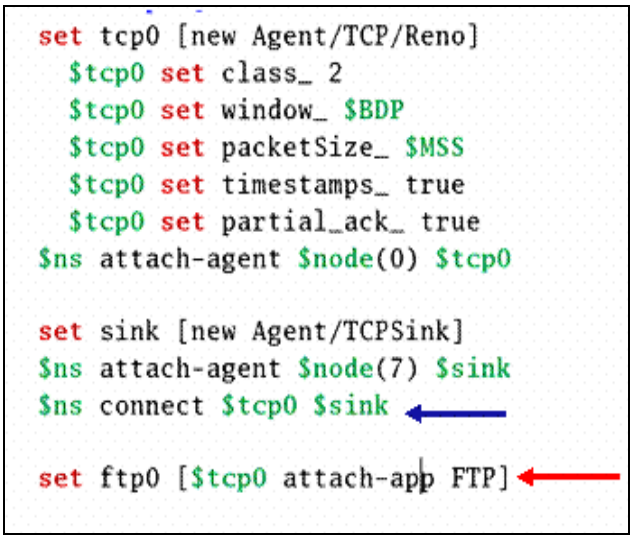

Figura 19: Criação de um agente TCP.

Assim como no *LanNode*, foram realizadas configurações nos parâmetros do agente TCP, os quais estão descritos na Tabela 2.

Tabela 2: Parâmetros do agente TCP.

Parâmetros	***Descrição***	***Valor***
class_	Define o agente padrão.	0
Window_	Tamanho máximo da janela.	3000
packetSize_	Tamanho do pacote que o agente injeta na rede.	1500

Até este ponto foram configurados e parametrizados parâmetros e *scripts* utilizados tanto na simulação de redes de alta e baixa velocidade. A partir de então existe a necessidade

de explorar as alterações nos parâmetros para prover a simulação com protocolos de transporte diferentes.

A Figura 19 apresenta qual deve ser a alteração realizada. O parâmetro *windowOption* está com o valor *highspeed*, caracterizando o algoritmo usado para controlar a janela de congestionamento.

Figura 20: Definição do algoritmo que controla a janela de congestionamento.

Além das simulações que geraram os resultados desse trabalho, outras duas foram utilizadas para aprendizado da linguagem Otcl. Elas estão relatadas nos apêndices III e IV. Para confecção desses *scripts* foi utilizado o pacote *ns-linux.patch* adquirido em (WEI, 2006). Através desse pacote é possível simular redes de alta velocidade utilizando diversos tipos de controle de congestionamento tais como: *Binary Increase Congestion control for TCP* (bic), H-TCP *congestion control* (htcp), TCP *Vegas congestion control* (vegas), *High Speed* TCP (HSTCP), dentre outros.

De acordo com Wei (2006) os resultados obtidos com utilização do pacote mencionado são satisfatórios, no entanto é direcionada para ambientes Linux. Por almejar resultados mais genéricos, o presente trabalho utilizou os controles de congestionamento oferecido no NS-2.

Feitas as adequações necessárias basta executar o *script* através do comando *run*. Os arquivos *.tcl* das simulações estão anexados a esse trabalho na integra.

4.2.2. Avaliação dos resultados

Foram analisados dois valores obtidos a partir de parâmetros do agente TCP, responsável por injetar pacotes na rede. São eles: cwnd (janela de congestionamento) e vazão. Este último foi obtido de acordo com a equação (1) (REZENDE, 2005).

$$\frac{\omega \times MSS}{RTT} \qquad \text{Equação (1)}$$

Para auxiliar na disposição gráfica dos resultados, dois arquivos foram utilizados para armazenamento dos valores: *resultados*, com valores de cwnd e *vazão*, com os resultados obtidos através da equação (1). A cada 0.5 tempos de simulação são capturados os valores. Isso é feito com o auxilio do procedimento chamado *monitor* (Figura 21), cuja ação é recursiva. Esse procedimento também foi realizado para o controle de congestionamento *highspeed*.

```
proc monitor {intervalo} {
        global ns tcp0
        set win [open resultadoTCP a]
        set win2 [open vazaoTCP a]
        puts $win  "[$tcp0 set cwnd_]"
        puts $win2 "[expr [$tcp0 set cwnd_] / [$tcp0 set rtt_]]"
        close $win
        close $win2

        $ns after $intervalo "monitor $intervalo"
}
```

Figura 21: Procedimento Monitor.

O próximo passo foi dispor os valores contidos nos arquivos em forma de gráfico, para melhor visualização. A seguir, são ilustrados os resultados.

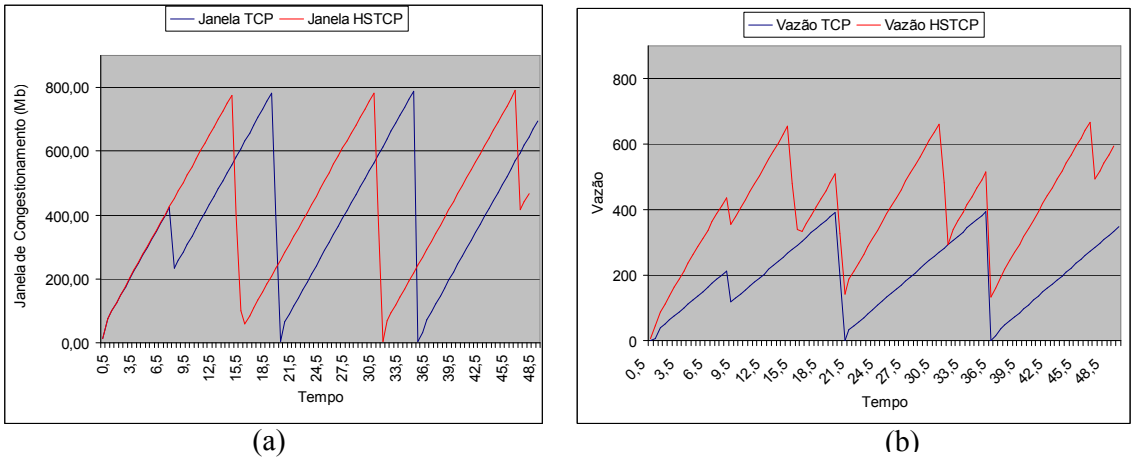

Figura 22: Desempenho utilizando 1Gb de banda na LAN.

Como pode ser observado na figura 22(a) as perdas de pacotes (simbolizadas pelos picos) são visíveis tanto na janela do TCP quanto na janela do HSTCP, porém nesse último o impacto sobre a vazão mostra-se menor (Figura 22 (b)). Embora possua mais pontos de queda, a vazão apresenta-se mais elevada durante todo o período de simulação quando utilizado controle *highspeed*.

Na Figura 23(a) a perda de seguimentos quando utilizado o controle de *highspeed* são mais acentuados, provocando quedas da janela de congestionamento mais freqüente quando comparada à janela do TCP. Porém esse menor desempenho em relação à janela não afeta a vazão. A utilização do controle *highspeed* propicia maior vazão à rede.

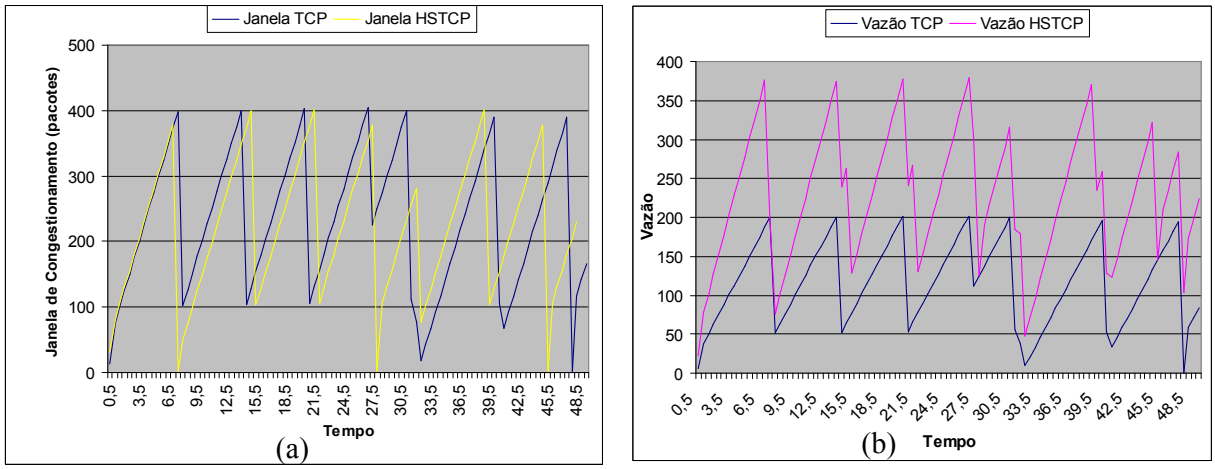

Figura 23: Desempenho utilizando 500Mb de banda entre o emissor e a LAN e 500Mb na LAN.

Nos gráficos apresentados é possível observar (como por exemplo, os instantes 6.7 e 27.5 na Figura 23(a)) o mecanismo de *slow start,* um algoritmo baseado na observação de que a taxa utilizada para injetar novos pacotes na rede deve ser a mesma em que as confirmações (ACK) são enviadas pelo outro destino. O emissor começa transmitindo apenas um segmento (MSS - *maximum segment size*) e espera pelo ACK correspondente. Quando este ACK é recebido, a janela de congestionamento é incrementada de um para dois, e dois segmentos podem ser mandados. Quando cada um destes dois segmentos for confirmado, a janela de congestionamento é aumentada para quatro. Isto provê um aumento exponencial. Logo em seguida o algoritmo *congestion avoidance* é iniciado tornando o crescimento da janela linear.

Outro algoritmo observado é o *fast recovery* (como por exemplo, instante 9.0 na Figura 22(a)), baseado na idéia de que se três ou mais ACKS duplicados forem recebidos em seguida, é um forte indício da perda de um segmento. O TCP realiza, então, a retransmissão imediata do segmento aparentemente perdido**,** sem esperar o cronômetro de retransmissão expirar (*timeout*). Este é o algoritmo de *fast retransmit*. Em seguida, *congestion avoidance*, e não *slow start*, é feito. Todo esse mecanismo compreende ao *fast recovery.*

4.3. Considerações finais

O presente Capítulo apresentou a ferramenta NS-2 e mostrou sua eficiência no estudo de controle de congestionamento em redes locais. Além disso, abordou os resultados obtidos bem como suas características. A partir do contexto explorado é possível extrair uma conclusão pertinente.

CONCLUSÕES

De acordo com os experimentos realizados e os resultados obtidos é possível concluir que o controle de congestionamento *highspeed* demonstra melhor desempenho em relação ao TCP Reno. Essa diferença é observada, principalmente, na vazão obtida pela rede, a qual se demonstra com taxas mais elevadas quando utilizado controle *highspeed*. Isso ocorre, poque a função de resposta do *highspeed* é mantida igual ao do TCP até uma determinada taxa de perda de segmentos, de modo que a janela possua um valor maior que no TCP.

Além disso, a eficiência e usabilidade da ferramenta *Network Simulator* é um ponto crucial na realização do presente trabalho. São disponibilizadas inúmeras funções para simular diversos tipos de redes com ou sem fio. Isso reforça a afirmação de que o NS é a ferramenta mais utilizada no meio acadêmico (COUTINHO, 2003).

TRABALHOS FUTUROS

Como possível proposta de continuação do trabalho tem-se a avaliação dos protocolos estudados em um ambiente híbrido com diversos agentes injetores de pacotes na rede.

DIFICULDADES ENCONTRADAS

A primeira dificuldade encontrada no presente trabalho foi referente à instalação da ferramenta *Network Simulator*, uma vez que as compilações baseadas no sistema operacional *Debian* (*Kurumim* e *Madriva*) não possuem todos os pacotes necessários para a instalação, sendo assim o sistema operacional utilizado foi o Fedora (compilação do Red Hat).

Outra dificuldade foi a confecção dos *scripts* para dispor os nós da rede simulada seguindo a topologia estrela, com um elemento concentrador o qual tivesse características de roteador. Isso foi resolvido com a utilização do elemento *LanNode.*

PUBLICAÇÕES

FERES, M. M., LUCAS, A. S., BRANCO, K. R. L. J. C.. **Avaliação do Desempenho do Protocolo TCP/IP em Redes com Topologia Estrela Utilizando a Ferramenta Network Simulator** In: XIV Congresso de Iniciação Científica da UFSCar, 2006, São Carlos. Anais de Eventos da Ufscar 2006. , 2006. v.2.

FERES, M. M., LUCAS, A. S., BRANCO, K. R. L. J. C.. **Avaliação de Desempenho do Protocolo TCP com Controle de Congestionamento Highspeed em Redes com Topologia Estrela**, 2006. 14º Simpósio Internacional de Iniciação Científica da USP, 2006, São Paulo.

REFERÊNCIAS

(BLESLAU *et. al.*, 2000) BRESLAU, Lee; ESTRIN, Deborah; FALL, Kevin; FLOYD, Sally; HEIDMANN, John; HELMY, Ahmed; HUANG, Polly; MCCANNE, Steven; VARADHAN, Kannan; XU, Ya; YU, Haobo. **Advances in Network Simulation**, Disponível na Internet em: http://www.isi.edu/~johnh/PAPERS/Breslau00a.pdf . Acesso em: 31 janeiro 2006.

(BRADEN, 1989) BREADEN, Robert. **Requirements for Internet Hosts -- Communication Layers**. USC/Information Sciences Institute. Disponível em: ftp://ftp.rfc-editor.org/in-notes/rfc1122.txt . Acesso em: 5 junho 2006.

(CABRITA, 2001) CABRITA, Jorge. Museu da Eletricidade. Site criado em Janeiro de 2001 Versão 3.1. Disponível na Internet em: http://geocities.yahoo.com.br/jcc5001pt/museutelegrafo.htm. Acesso em 22 março de 2006.

(COLCHER *et. al.*, 2004) COLCHER, Sérgio; LEMOS, Guido; SOARES, Luiz Fernando Gomes. . Redes de computadores: das LANs MANs e WANs às redes ATM. 2ª ed. Rio de Janeiro: Campus, 2004.

(CNET, 2002) Manual do simulador de redes *Cnet.* Disponível na Internet em: http://www.csse.uwa.edu.au/cnet/ . Acesso em: 6 fevereiro 2006.

(COMER, 2005) COMER, Douglas E. . **Interligação em rede com TCP/IP: princípios, protocolos e arquitetura**. Rio de Janeiro: Campus, 2005.

(COMER, 1999) COMER, Douglas E. . **Interligação em rede com TCP/IP: princípios, protocolos e arquitetura**. Volume II. Rio de Janeiro: Campus, 1999.

(COUTINHO, 2003) COUTINHO, Mauro M. . "Network Simulator Guia Básico para Iniciantes". Universidade Federal do Pará – UFPa. Agosto 2003. Disponível na Internet em: http://www.cci.unama.br/margalho/networksimulator/pdfs/nsr1.pdf . Acesso em: 30 janeiro 2006.

(CYCLADES BRASIL, 2001) Cyclades Brasil, **Guia Internet de conectividade/ Cyclades** Brasil. – 8ª ed.- São Paulo : Editora SENAC São Paulo, 2001.

(CHUNG *et .al.*, 2003) CHUNG, Jae; CLAYPOOL, Mark**. NS by Example** . Disponível na Internet em: http://nile.wpi.edu/NS/ . Acesso em: 14 novembro 2006.

(DANTAS, 2005)DANTAS, Mario. **Computação Distribuída de Alto Desempenho: Redes, Clusters e Grids Computacionais**. Rio de Janeiro: Editora Axel, 2005.

(DIAS *et.al.,* 2003) DIAS, Paolla de Souza; FIQUEIREDO, Valeska dos Reis; ABRÃO, Iran Calixto; CORREIA, Cláudio. **Guia de Uso do *Software* OpNet**. Poços de Caldas: agosto 2003. Relatório Técnico, Pontifícia Universidade Católica de Minas Gerais.

(FALL, 2006) FALL, K.; Varadhan, K.. **The NS Manual**; Network Simulator 2, VINT Project; 2002. Disponível em: http://www.isi.edu/nsnam/ns/ . Acesso em: 2 agosto 2006.

(FLOYD, 2003) FLOYD Sally, **HighSpeed TCP for Large Congestion Windows**. RFC 3649, dezembro de 2003.

(GLOMOSIM 2003) **GloMoSim Web Site**. Disponível na Internet em http://pcl.cs.ucla.edu/projects/glomosim/, acesso em 8 de novembro de 2006.

(INFO, 2005) FORTES, Débora. A era das redes. Revista Info Exame. São Paulo, Ano 20, n. 237, p.46-48, dez. 2005.

(FALK et al, 2003) FALK Aaron, Faber Teg, Bannister Joseph, Chien Andrew, Grossman Robert e Leigh Jason, Transport procotols for high performance, Communications of the ACM, vol. 46, no. 11, pp. 43–49, novembro de 2003.

(JIN *et. al.,* 2003) JIN, Cheng; X., David; WEI, Steven; LOW, H., **FAST TCP: Motivation, architecture, algorithms, performance**, em IEEE INFOCOM, março de 2004.

(NWG, 1998) NETWORK WORKING GROUP, **Internet Protocol, Version 6 (IPv6) Specification.** Dezembro 1998. Disponível na Internet em: ftp://ftp.rfc-editor.org/in-notes/rfc2460.txt . Acesso em: 5 junho 2006.

(OLIVEIRA, 2002) OLIVEIRA, Igor Pereira Marciano de; CASTRO, Leonardo Ferreira de.. **Estudo do Comportamento do TCP/IP em Enlaces Via Satélite**. Projeto Final de Graduação, Publicação G12/2002, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília-DF, 113p. Disponível na Internet em: https://www.redes.unb.br/Biblio_diss/vrml/pdf/UnB%20LabRedes%20PFG%202002%2012.pdf . Acesso em 2 fevereiro 2006.

(QNET, 2004) Manual QNET Disponível na Internet em:

http://www.cin.ufpe.br/~cak/publications/sbrc2004_QNET.pdf . Acesso em: 6 fevereiro 2006.

(OpNet, 2006). OPNET – **Making Networks and Applications Perform**. Disponível na Internet em http://www.opnet.com/, acesso em 9 de novembro de 2006.

(OTCL, 1995) OTCL Tutorial. Disponível na Internet em: ftp://ftp.tns.lcs.mit.edu/pub/otcl/doc/tutorial.html. Acesso em: 6 fevereiro 2006.

(ANDRADE *et. al.,* 2003) ANDRADE, Rogério; KAMIENSKI, Carlos; SOUSA, Dênio; SADOK Djamel. **O Algoritmo SQM-Response para Controle de Congestionamento do Protocolo TCP**. 21º Simpósio Brasileiro de Redes de Computadores (SBRC 2003), Natal/RN, Maio 2003.

(REZENDE, 2005) REZENDE, José Ferreira de ; COSTA, Luís Henrique Maciel Kosmalski ; RUBINSTEIN, M. G. . **Avaliação Experimental e Simulação do Protocolo TCP em Redes de Alta Velocidade**. In: XXI Simpósio Brasileiro de Telecomunicações, 2005, Campinas. Anais do XXI Simpósio Brasileiro de Telecomunicações, 2005. p. 53-58.

(STALLINGS, 2005) STALLINGS, William. **Redes de Sistemas de Comunicação de Dados**. Ed. Elsevier Editora Ltda., 2005.

(TANENBAUM, 1997) TANENBAUM, Andrew S. **Redes de Computadores.** 3ª ed Rio de Janeiro: Editora Campus, 1997.

(TANENBAUM, 2003) TANENBAUM, Andrew S**. Redes de Computadores**. 4. ed. Rio de Janeiro: Editora Campus, 2003.

(TORRES, 2001) TORRES, Gabriel. **Redes de Computadores – Curso Completo**. Editora Axcel Books, Rio de Janeiro, 2001.

(VINT, 1997) VINT. Site oficial do projeto VIIN. Disponível na Internet em: http://www.isi.edu/nsnam/vint/ . Acesso em 17 novembro 2006.

(WEI, 2006) WEI, David X.; CAO, Pei. **NS2 TCPLinux: An NS2 TCP Implementation with Congestion Control Algorithms from Linux**. Workshop of NS-2, 2006. Disponível na Internet em: wttp://www.cs.caltech.edu/~weixl/technical/ns2linux/index.html. Acesso em: 5 agosto 2006.

APÊNDICE A – *Script* para simulação de uma LAN com agente TCP/Reno.

```
# Autor: Mariana M. Feres
# Modificação de "ns-2/tcl/ex/lantest.tcl"

set MSS 1500
set BDP 30000

set opt(tr) "lan.tr"
set opt(namtr)    "lan.nam"
set opt(seed)     0
set opt(stop)     50
set opt(node)     8

set opt(qsize)    100
set opt(bw) 1000Mb
set opt(delay)    5ms
set opt(ll) LL
set opt(ifq)      Queue/DropTail
set opt(mac)      Mac/802_3
set opt(chan)     Channel
set opt(tcp)      TCP/Reno
set opt(sink)     TCPSink

set opt(app)      FTP


proc finish {} {
      global ns opt trfd ntrfd

      $ns flush-trace
#     close $trfd
        close $ntrfd
      exec nam $opt(namtr) &
      exit 0
}
proc monitor {intervalo} {
      global ns tcp0
      set win [open resultadoTCP a]
      set win2 [open vazaoTCP a]
      puts $win  "[$tcp0 set cwnd_]"
      puts $win2 "[expr [$tcp0 set cwnd_] / [$tcp0 set rtt_]]"
      close $win
      close $win2

      $ns after $intervalo "monitor $intervalo"
}
proc create-trace {} {
      global ns opt

      set trfd [open $opt(tr) w]
      #$ns create-trace Drop $trfd $node0 $node(7) ;#mexi aqui
      $ns trace-all $trfd
      return $trfd
}

proc create-namtrace {} {
        global ns opt

        set ntrfd [open $opt(namtr) w]
        $ns namtrace-all $ntrfd
```

```
}

proc create-topology {} {
      global ns opt node0 node1 BDP
      global lan node source node0 node1

      set num $opt(node)
      for {set i 0} {$i < $num} {incr i} {
            set node($i) [$ns node]
            lappend nodelist $node($i)
      }

      set lan [$ns newLan $nodelist $opt(bw) $opt(delay) \
                  -llType $opt(ll) -ifqType $opt(ifq) \
                  -macType $opt(mac) -chanType $opt(chan)]

      set node0 [$ns node]
      $ns duplex-link $node0 $node(0) 6000Mb 2ms DropTail
      $ns duplex-link-op $node0 $node(0) orient right
      $ns queue-limit $node0 $node(0) $BDP

#
}

## MAIN ##

set ns [new Simulator]

#set trfd [create-trace]
set ntrfd [create-namtrace]
create-topology

set tcp0 [new Agent/TCP/Reno]
  $tcp0 set class_ 1
  $tcp0 set window_ $BDP
  $tcp0 set packetSize_ $MSS
  $tcp0 set timestamps_ true
  $tcp0 set partial_ack_ true
$ns attach-agent $node(0) $tcp0

set sink [new Agent/TCPSink]
$ns attach-agent $node(7) $sink
$ns connect $tcp0 $sink

set ftp0 [$tcp0 attach-app FTP]


$ns at 0.5 "monitor 0.5"

$ns at 0.0 "$ftp0 start"
#$ns at 0.5 "$cbr start"
#$ns at 49.0 "$cbr stop"
$ns at $opt(stop) "finish"

$ns run
```

APÊNDICE B – *Script* para simulação de uma LAN com agente TCP e controle de congestionamento *highspeed*.

```
# Autor: Mariana M. Feres
# Modificação de "ns-2/tcl/ex/lantest.tcl"

set MSS 1500
set BDP 30000

set opt(tr) "lan.tr"
set opt(namtr)    "lan.nam"
set opt(seed)     0
set opt(stop)     50
set opt(node)     8

set opt(qsize)    100
set opt(bw) 1000Mb
set opt(delay)    5ms
set opt(ll) LL
set opt(ifq)      Queue/DropTail
set opt(mac)      Mac/802_3
set opt(chan)     Channel
set opt(tcp)      TCP/Reno
set opt(sink)     TCPSink

set opt(app)      FTP


proc finish {} {
      global ns opt trfd ntrfd

      $ns flush-trace
#     close $trfd
        close $ntrfd
      exec nam $opt(namtr) &
      exit 0
}
proc monitor {intervalo} {
      global ns tcp0
      set win [open resultadoTCP a]
      set win2 [open vazaoTCP a]
      puts $win  "[$tcp0 set cwnd_]"
      puts $win2 "[expr [$tcp0 set cwnd_] / [$tcp0 set rtt_]]"
      close $win
      close $win2

      $ns after $intervalo "monitor $intervalo"
}
proc create-trace {} {
      global ns opt

      set trfd [open $opt(tr) w]
      #$ns create-trace Drop $trfd $node0 $node(7) ;#mexi aqui
      $ns trace-all $trfd
      return $trfd
}

proc create-namtrace {} {
        global ns opt

        set ntrfd [open $opt(namtr) w]
```

```
        $ns namtrace-all $ntrfd
}

proc create-topology {} {
      global ns opt node0 node1 BDP
      global lan node source node0 node1

      set num $opt(node)
      for {set i 0} {$i < $num} {incr i} {
            set node($i) [$ns node]
            lappend nodelist $node($i)
      }

      set lan [$ns newLan $nodelist $opt(bw) $opt(delay) \
                  -llType $opt(ll) -ifqType $opt(ifq) \
                  -macType $opt(mac) -chanType $opt(chan)]

      set node0 [$ns node]
      $ns duplex-link $node0 $node(0) 6000Mb 2ms DropTail
      $ns duplex-link-op $node0 $node(0) orient right
      $ns queue-limit $node0 $node(0) $BDP
}

## MAIN ##

set ns [new Simulator]

#set trfd [create-trace]
set ntrfd [create-namtrace]
create-topology

set tcp0 [new Agent/TCP]
  $tcp0 set class_ 1
  $tcp0 set window_ $BDP
  $tcp0 set packetSize_ $MSS
  $tcp0 set timestamps_ true
  $tcp0 set partial_ack_ true
  $tcp0 set windowOption_ highspeed

$ns attach-agent $node(0) $tcp0

set sink [new Agent/TCPSink]
$ns attach-agent $node(7) $sink
$ns connect $tcp0 $sink

set ftp0 [$tcp0 attach-app FTP]


$ns at 0.5 "monitor 0.5"

$ns at 0.0 "$ftp0 start"
#$ns at 0.5 "$cbr start"
#$ns at 49.0 "$cbr stop"
$ns at $opt(stop) "finish"

$ns run
```

APÊNDICE C – *Script* para simulação de uma rede com quarto nós utilizando o pacote TCP-Linux e com controle de congestionamento *highspeed.*

```
set MSS 1500        ;#máximo tamanho do segmento
set BDP 3000       ;#tamanho maximo do buffer da fila entre dois nós
set MainBW "10000Mb" ;#fast ethernet, banda do receptor
set SideBW "10000Mb" ;#bw entre o servidor e o roteador da rede estrela
set SideDelay "5ms"
set MainDelay "50ms"
set MainBuffer 2000

#Create a simulator object
set ns [new Simulator]

#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

#Open the NAM trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#Define a 'finish' procedure
proc finish {} {
        global ns nf
        $ns flush-trace
        #Close the NAM trace file
        close $nf
        #Execute NAM on the trace file
        exec nam out.nam &
        exit 0
}

proc monitor {interval} {
    global tcp2 tcp ns rcvr
    set nowtime [$ns now]
  set win [open result1 a]
    set win2 [open result2 a]
    set win3 [open result3 a]
    set win4 [open result4 a]
        #janela de congestionamento    maior valor do reconhecimento
    puts $win  "[$tcp set cwnd_]"
    puts $win2 "[$tcp set awnd_]"
    puts $win3 "[$tcp2 set cwnd_]"
    puts $win4 "[$tcp2 set awnd_]"
    close $win
    close $win2
    close $win3
    close $win4
    $ns after $interval "monitor $interval"
}

#Criação dos nós
set n0 [$ns node]
$n0 shape box
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

#Create links between the nodes
$ns duplex-link $n0 $n2 $SideBW $SideDelay DropTail
```

```
$ns duplex-link $n1 $n2 $SideBW $SideDelay DropTail
$ns duplex-link $n2 $n3 $MainBW $MainDelay  DropTail

#Set Queue Size of link (n2-n3)
$ns queue-limit $n2 $n3 $MainBuffer

#Give node position (for NAM)
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right

#Monitor the queue for link (n2-n3). (for NAM)
$ns duplex-link-op $n2 $n3 queuePos 0.5

#Setup a TCP connection
set tcp [new Agent/TCP/Linux]
#$tcp set class_ 2
 $tcp set packetSize_ $MSS
 $tcp set window_ 90000
 $tcp set timestamps_ true
 $tcp set partial_ack_ true
 $tcp set rtt_ 100 ;#roud trip time
#$tcp set windowOption_ highspeed
$ns at 0 "$tcp select_ca highspeed"
$ns attach-agent $n0 $tcp

set tcp2 [new Agent/TCP/Fack]
 $tcp2 set class_ 2
 $tcp2 set packetSize_ $MSS
 $tcp2 set window_ 90000
 $tcp2 set timestamps_ true
 $tcp2 set partial_ack_ true
 $tcp set rtt_ 100
# $tcp2 set windowOption_ highspeed
#$ns at 0 "$tcp2 select_ca highspeed"
$ns attach-agent $n1 $tcp2

set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
$ns connect $tcp $sink
$tcp set fid_ 2

set sink2 [new Agent/TCPSink]
$ns attach-agent $n3 $sink2
$ns connect $tcp2 $sink2
$tcp2 set fid_ 1

#Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP

set ftp2 [new Application/FTP]
$ftp2 attach-agent $tcp2
$ftp set type_ FTP

#Schedule events for the CBR and FTP agents
$ns at 0.1 "$ftp start"
$ns at 1.0 "$ftp2 start"
$ns at 49.0 "$ftp2 stop"
$ns at 50.0 "$ftp stop"
```

```
#Detach tcp and sink agents (not really necessary)
$ns at 50.0 "$ns detach-agent $n0 $tcp ; $ns detach-agent $n3 $sink"

$ns at 0 "monitor 0.5"

#Call the finish procedure after 5 seconds of simulation time
$ns at 50.0 "finish"

#Run the simulation
$ns run
```

APÊNDICE D – *Script* para simulação de uma rede com quarto nós utilizando o pacote TCP-Linux e com controle de congestionamento highspeed.

```
set MSS 1500        ;#máximo tamanho do segmento
set BDP 3000       ;#tamanho maximo do buffer da fila entre dois nós
set MainBW "6000Mb" ;#fast ethernet, banda do receptor
set SideBW "6000Mb" ;#bw entre o servidor e o roteador da rede estrela
set SideDelay "5ms"
set MainDelay "50ms"
set FlowNumber 1
set MainBuffer 2000

#Create a simulator object
set ns [new Simulator]

#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

#Open the NAM trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#Define a 'finish' procedure
proc finish {} {
        global ns nf
        $ns flush-trace
        #Close the NAM trace file
        close $nf
        #Execute NAM on the trace file
        exec nam out.nam &
        exit 0
}

proc monitor {interval} {
    global tcp2 tcp ns rcvr
    set nowtime [$ns now]
    set win [open result1 a]
    set win2 [open result2 a]
    set win3 [open result3 a]
    set win4 [open result4 a]
        #janela de congestionamento    maior valor do reconhecimento
    puts $win  "[$tcp set cwnd_]"
    puts $win2 "[$tcp set awnd_]"
    puts $win3 "[$tcp2 set cwnd_]"
    puts $win4 "[$tcp2 set awnd_]"
    close $win
    close $win2
    close $win3
    close $win4
    $ns after $interval "monitor $interval"
}

#Criação dos nós
set n0 [$ns node]
$n0 shape box
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

#Create links between the nodes
```

```
$ns duplex-link $n0 $n2 $SideBW $SideDelay DropTail
$ns duplex-link $n1 $n2 $SideBW $SideDelay DropTail
$ns duplex-link $n2 $n3 $MainBW $MainDelay  DropTail

#Set Queue Size of link (n2-n3) to 10
$ns queue-limit $n2 $n3 $MainBuffer

#Give node position (for NAM)
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right

#Monitor the queue for link (n2-n3). (for NAM)
$ns duplex-link-op $n2 $n3 queuePos 0.5

#Setup a TCP connection
set tcp [new Agent/TCP/Fack]
 $tcp set class_ 2
 $tcp set packetSize_ $MSS
 $tcp set window_ 90000
 $tcp set timestamps_ true
 $tcp set partial_ack_ true
$ns attach-agent $n0 $tcp

set tcp2 [new Agent/TCP/Fack]
 $tcp2 set class_ 2
 $tcp2 set packetSize_ $MSS
 $tcp2 set window_ 90000
 $tcp2 set timestamps_ true
 $tcp2 set partial_ack_ true
$ns attach-agent $n1 $tcp2

set sink [new Agent/TCPSink/Sack1]
$ns attach-agent $n3 $sink
$ns connect $tcp $sink
$ns connect $tcp2 $sink
$tcp set fid_ 2

#set sink2 [new Agent/TCPSink/Sack1]
#$ns attach-agent $n3 $sink2
#$ns connect $tcp2 $sink2
#$tcp2 set fid_ 1

#Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP

set ftp2 [new Application/FTP]
$ftp2 attach-agent $tcp2
$ftp set type_ FTP

#Schedule events for the CBR and FTP agents
$ns at 0.1 "$ftp start"
$ns at 1.0 "$ftp2 start"
$ns at 49.0 "$ftp2 stop"
$ns at 50.0 "$ftp stop"

#Detach tcp and sink agents (not really necessary)
$ns at 50.0 "$ns detach-agent $n0 $tcp ; $ns detach-agent $n3 $sink"
```

```
$ns at 0 "monitor 0.5"

#Call the finish procedure after 5 seconds of simulation time
$ns at 50.0 "finish"

#Run the simulation
$ns run
```