UNIVERSIDADE FEEVALE

MÁRCIO DIEGO BERLITZ

SCTP: UM ESTUDO TEÓRICO E EXPERIMENTAL UTILIZANDO A FERRAMENTA NS2

Novo Hamburgo

2014

MÁRCIO DIEGO BERLITZ

SCTP: UM ESTUDO TEÓRICO E EXPERIMENTAL UTILIZANDO A FERRAMENTA NS2

Trabalho de Conclusão de Curso
apresentado como requisito parcial
à obtenção do grau de Bacharel em
Ciência da Computação pela
Universidade Feevale

Orientador: Vandersilvio da Silva

Novo Hamburgo

2014

AGRADECIMENTOS

Gostaria de agradecer a todos os que, de alguma maneira, contribuíram para a realização desse trabalho de conclusão, em especial:

A minha família, que sempre esteve presente nos momentos mais difíceis, em especial, a minha mãe que sempre me apoiou.

Aos meus amigos, que sempre estiveram presentes, em especial aos que conquistei ao longo destes anos em que estive na Feevale.

Agradeço ainda, aos professores da Feevale, com os quais tive o prazer de conviver durante todos esses anos. Em especial ao meu orientador, o professor Vandersilvio da Silva, que me guiou durante todo este trabalho.

RESUMO

Hoje em dia as redes de computadores são encontradas em todos os lugares desde uma residência familiar até grandes empresas multinacionais, isso acontece devido sua grande versatilidade, pois ela pode ser utilizada por uma grande variedade de aplicações que são capazes de atender as necessidades de cada usuário. Atualmente os protocolos de transporte mais utilizados são o UDP e o TCP. O protocolo UDP oferece transporte rápido, porém não confiável. Já o protocolo TCP tem como objetivo principal oferecer a confiabilidade e controle não existente na camada de rede da Internet. Apesar de ser um protocolo confiável, um número crescente de aplicações recentes tem descoberto que o TCP é muito limitante. Com a evolução da tecnologia e a necessidade de obter melhores resultados surgiu a necessidade de novos protocolos, como o SCTP. O STCP foi elaborado visando o atendimento de requisitos de transmissão de sinalização telefônica sobre redes IP. Contudo, as características do SCTP o tornam atraente para inúmeras aplicações da Internet, em oposição aos protocolos tradicionalmente utilizados. A proposta deste trabalho é apresentar as funcionalidades do SCTP com aquelas ligadas ao protocolo TCP, que é o protocolo de transporte confiável da Internet. Desta forma, pretende-se não apenas mostrar as características do protocolo SCTP, mas também realizar um estudo comparativo dos protocolos SCTP e TCP através de ferramentas de simulação de redes.

Palavras-chave: Protocolos de Transporte. SCTP. Internet. Redes.

ABSTRACT

Nowadays computer networks are found everywhere from a family residence to large multinational companies, this happens because of its versatility, as it can be used by a wide variety of applications that are able to meet the needs of each user. Currently the most used transport protocols are TCP and UDP. UDP provides fast shipping, but not reliable. Have the TCP protocol aims to provide reliable and not existing control in the Internet network layer. Despite being a reliable protocol, a growing number of recent applications have found that TCP is too limiting. With the evolution of technology and the need for best results the need for new protocols emerged, such as SCTP. STCP was developed aiming at meeting the reporting requirements of telephony signaling over IP networks. However, the features of SCTP make it attractive for many applications of the Internet, as opposed to the traditionally used protocols. The purpose of this paper is to present the features of SCTP with those connected to the TCP protocol, which is a reliable transport protocol of the Internet. Thus, we intend to not only show the characteristics of the SCTP protocol, but also perform a comparative study of SCTP and TCP protocols through network simulation tools.

Key words: Transport Protocols. SCTP. Internet. Networks.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ASCII	American Standard Code for Information Interchange
DARPA	Defense Advanced Research Projects Agency
HTTP	HyperText Transfer Protocol
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISO	International Standards Organization
MAC	Message Authentication Code
MTU	Maximum Transmission Unit
MSS	Maximum Segment Size
NS	Network Simulator
OSI	Open Systems Interconnection
OTcL	Object Tool Command Language
RFC	Request for Comments
SACK	Selective Acknowledgement
SCTP	Stream Control Transmission Protocol
SIGTRAN	Signalling Transport
TCP	Transport Control Protocol
UDP	User Datagram Protocol
VINT	Virtual InterNetwork Testbed

SUMÁRIO

INTRODUÇÃO

A Internet tem revolucionado muitos aspectos em nossas vidas. Ela tem afetado a maneira como fazemos negócios, assim como o modo de passarmos o nosso tempo livre. A Internet trouxe benefícios na utilização das tecnologias com fácil acesso ao conhecimento, na colaboração entre as pessoas e organizações, na inclusão social, e na criação de valores.

O grande objetivo das redes, e o que a torna tão importante é ela fazer com que múltiplos usuários em distâncias indeterminadas compartilhem um determinado recurso. Tanenbaum (2011, p.19) define o objetivo das redes de computadores como "tornar todos os programas, equipamentos e especialmente dados ao alcance de todas as pessoas na rede, independentemente da localização física do recurso e do usuário".

A Internet consiste em um conjunto de padrões (protocolos) que definem como as informações são transmitidas pela rede. Sendo um conjunto de protocolos, ou pilha de protocolos, a Internet oferece serviços de comunicação de forma modular, tratando problemas específicos em etapas bem definidas. Devido a essa característica multiprotocolar, a Internet é usualmente denominada de pilha TCP/IP, em alusão a dois de seus protocolos: o IP (*Internet Protocol*) e o TCP (*Transport Control Protocol*) (COSTA 2005).

Atualmente nas pilhas de protocolos de rede típicas, como TCP/IP, a confiabilidade na entrega dos dados não é garantida, devido à natureza do protocolo IP. Dessa forma, algum mecanismo deve ser fornecido para garantir a confiabilidade na transmissão dos dados, caso esse serviço seja necessário. Para oferecer a confiabilidade necessária na comunicação em rede, é necessário a utilização de protocolos de transporte (COSTA, 2005).

O principal objetivo dos protocolos de transporte é oferecer um serviço confiável, eficiente e econômico a seus usuários. São responsáveis por promover uma transferência de dados confiável e econômica entre máquina de origem e máquina de destino, independente das redes físicas em uso no momento (TANENBAUM, 2011). Os protocolos mais conhecidos são o UDP (*User Datagram Protocol*) (DARPA, 1980) e o TCP (DARPA, 1981).

Segundo Costa (2005), a Internet originalmente definiu os protocolos UDP e TCP como padrões para transmissão dados, chamados de protocolos de transporte. As aplicações que desejam uma comunicação rápida e flexível recorrem ao protocolo UDP, enquanto o TCP é destinado às comunicações confiáveis. Entretanto, algumas aplicações precisam de confiabilidade, porém, são limitadas ou tornam-se menos eficientes devido à operação do TCP.

De acordo com Stewart et al. (2000) o TCP tem realizado imenso serviço como o principal meio de transferência confiável de dados em redes IP. No entanto, um número crescente de aplicações recentes descobriram TCP muito limitante, e assim incorporaram seu próprio protocolo de transferência de dados confiável sobre UDP.

Com a evolução das aplicações e da Internet, e a necessidade de obter melhores resultados em determinadas situações, surgiu à necessidade de utilização de novos protocolos, como o SCTP (*Stream Control Transmission Protocol*). Conforme Costa (2005) o SCTP foi inicialmente elaborado visando o atendimento de requisitos de transmissão de sinalização telefônica sobre redes IP. Contudo, as características do SCTP o tornam atraente para inúmeras aplicações da Internet, em oposição aos protocolos mais tradicionais.

Segundo Stewart et al. (2000) o protocolo SCTP é um protocolo de transporte confiável que opera em cima de uma rede de pacotes sem conexão, como o IP. Costa (2005) afirma que o protocolo SCTP apresenta todas as características de um protocolo de transporte confiável, como o TCP, porém, possui funcionalidades adicionais em relação a este último, conhecidas como *multistreaming* e *multihoming*. De acordo com Stewart e Metz (2001, tradução nossa) o SCTP apresenta algumas características herdadas do TCP, mas oferece outras funcionalidades adicionais como:

• Entrega confirmada, livre de erros e não duplicada de dados de usuário;

• Fragmentação de dados em conformidade com o MTU (*Maximum Transmission Unit*) descoberto do caminho;

• SCTP suporta vários modos de entrega incluindo ordem de transmissão estrita (como o TCP), parcialmente ordenado e entrega desordenada (como o UDP).

• Entrega sequencial de mensagens de usuário em múltiplos fluxos, com opção para entrega por ordem de chegada de mensagens de usuário individuais;

• Empacotamento opcional de múltiplas mensagens de usuário em um único pacote SCTP;

• Tolerância a falhas de rede através do suporte a multicaminhos (*multihoming),* SCTP envia pacotes para um endereço IP de destino, mas pode redirecionar mensagens para um suplente se o endereço IP atual torna-se inacessível.

O protocolo UDP é considerado um protocolo não confiável, já o protocolo TCP apesar de ser confiável, tem-se apresentado limitado e menos eficiente em certas ocasiões. Com a crescente necessidade de melhores resultados, e a criação de novos protocolos de transporte, como o SCTP, surge à necessidade da presente pesquisa. Fazer um estudo comparativo do protocolo SCTP com o protocolo TCP, que é o protocolo mais utilizado na atualidade. Propõe-se então, apresentar características do protocolo SCTP, estudar a aplicabilidade do mesmo, bem como realizar testes de vazão e latência utilizando ferramentas de simulação de redes.

1 MODELOS DE REFERÊNCIA

Nas duas próximas seções serão examinadas duas importantes arquiteturas de rede: os modelos de referência OSI (*Open Systems Interconnection*) e TCP/IP, junção do *Transmision Control Protocol* com o *Internet Protocol*. Segundo Tanenbaum (2011) os protocolos associados ao modelo OSI raramente são utilizados nos dias de hoje, o modelo em si é de fato bastante geral e ainda válido, e as características descritas em cada camada ainda são muito importantes. O modelo TCP/IP tem características opostas, o modelo propriamente dito não é muito utilizado, mas os protocolos são bastante utilizados.

1.1 MODELO OSI

Quando as redes de computadores surgiram, as soluções eram, na maioria das vezes, proprietárias, ou seja, uma determinada tecnologia só era suportada por seu fabricante. Não havia a possibilidade de se misturar soluções de fabricantes diferentes. Dessa forma, um mesmo fabricante era responsável por construir praticamente tudo na mesma rede (TORRES, 2007).

Segundo com Dantas (2005) nenhum grande fabricante tinha interesse em ter interoperabilidade com outros ambientes computacionais em termos de hardware e software. Por outro lado, os usuários e as organizações tinham a cada dia seus sistemas mais heterogêneos, sem a possibilidade de compartilhamento efetivo dos seus recursos.

"Para facilitar a interconexão de sistemas de computadores, a ISO (*International Standards Organization*) desenvolveu um modelo de referência chamada OSI para que os fabricantes pudessem criar protocolos a partir deste modelo" (TORRES, 2007, p.1). Segundo Torres (2007) protocolo se trata de uma "linguagem" usada para transmitir dados pela rede. Para que dois computadores possam se comunicar, eles devem usar o mesmo protocolo, ou seja, a mesma linguagem.

O objetivo do modelo OSI é mostrar que é possível facilitar a comunicação entre sistemas heterogêneos, sem exigir a alteração na lógica de hardware e/ou software. O modelo OSI é uma estrutura em camadas para projetos de sistemas de rede que permite a comunicação entre todos os tipos de sistemas de computador. Ele consiste em sete camadas separadas, porém relacionadas, cada uma definindo uma parte do processo de movimentação da informação por uma rede (FOROUZAN e FEGAN, 2009).

1.1.1 Camada Física

Esta é a primeira camada do modelo, "seu objetivo é a definição elétrica e mecânica da interface de rede" (DANTAS, 2002, p.114). De acordo com Tanenbaum (2011) as principais questões que devem ser consideradas são quais os sinais elétricos que devem ser usados para representar um bit 1 e um bit 0, o tempo de duração dos bits, se a transmissão pode ou não ser realizada simultaneamente nos dois sentidos, a forma como a conexão inicial será estabelecida e de que maneira ela será encerrada quando ambos os lados tiverem terminado, e ainda quantos pinos o conector de rede terá e qual será a finalidade de cada pino.

1.1.2 Camada de Enlace de Dados

A principal função desta camada é transformar um canal de transmissão normal em uma linha que pareça livre de erros de transmissão. Para fazer isso, a camada de enlace "mascara" os erros, de modo que a camada de rede não os veja (TANENBAUM, 2011).

A camada de enlace pega os pacotes de dados recebidos da camada de rede e os transforma em quadros de dados que serão trafegados pela rede, acrescentando informações como o endereço da placa de rede de origem, o endereço da placa de rede de destino, dados de controle, os dados em si e uma soma de verificação. Quando o receptor recebe um quadro, a sua camada de enlace verifica se o dado chegou íntegro. Se os dados estiverem íntegros, o receptor envia de volta um quadro de confirmação. Caso essa confirmação não seja recebida, a camada de enlace do transmissor vai reenviar o quadro, já que ele não chegou até o receptor ou então os dados chegaram corrompidos (TORRES, 2007).

Como a função da camada 1 é apenas física (elétrica e mecânica), a camada de enlace é responsável pelo reconhecimento do início e final dos quadros, pelo controle de fluxo entre remetente e destinatário e ainda pela forma de acesso ao meio (DANTAS, 2002).

1.1.3 Camada de Rede

Segundo Torres (2007) a camada de rede é responsável pelo endereçamento dos pacotes, convertendo endereços lógicos em endereços físicos, de forma que os pacotes consigam chegar

corretamente ao seu destino. Essa camada também determina a rota que os pacotes vão seguir até chegaram ao seu destino, levando em consideração condições de tráfego da rede.

Se houver muitos pacotes na rede ao mesmo tempo, eles vão dividir o mesmo caminho, formando possíveis gargalos. A camada de rede em conjunto com as camadas mais altas é responsável pelo controle desse congestionamento. Outros serviços de qualidade como atraso, tempo em trânsito e instabilidade também são uma questão da camada de rede (TANENBAUM, 2011).

Quando um pacote é enviado de uma rede para outra, podem surgir muitos problemas até o pacote chegar ao seu destino. O endereçamento utilizado por uma das redes pode ser diferente, a rede de destino talvez não aceite o pacote por ele ser muito grande, os protocolos podem ser diferentes, entre outros possíveis problemas. Cabe a camada de rede superar todos esses problemas, a fim de permitir que redes heterogêneas sejam interconectadas (TANENBAUM, 2011).

1.1.4 Camada de Transporte

A camada de transporte é responsável por pegar os pacotes recebidos da camada de sessão e dividi-los em unidades menores, se for necessário, e repassar essas unidades à camada de rede. No computador receptor, a camada de transporte é responsável por pegar os pacotes recebidos da camada de rede e remontar o dado original para enviá-lo para à camada de sessão. Isso inclui o controle de fluxo (colocar os pacotes recebidos em ordem, caso tenham chegado fora de ordem) e correção de erros, geralmente enviando para o transmissor uma informação de reconhecimento, informando que o pacote foi recebido com sucesso (TORRES, 2007).

1.1.5 Camada de Sessão

De acordo com Torres (2007) a camada de sessão permite que dois aplicativos em diferentes máquinas estabeleçam sessões de comunicação. Segundo Tanenbaum (2011), uma sessão oferece diversos serviços, como o controle de diálogo (que permite controlar quem pode transmitir em cada momento), o gerenciamento de *tokens* (que impede que duas partes tentem executar a mesma operação crítica ao mesmo tempo) e a sincronização (que realiza a verificação

periódica de transmissões para permitir que elas continuem a partir do ponto que estavam ao ocorrer uma falha).

1.1.6 Camada de Apresentação

Segundo Torres (2007) a camada de apresentação também é conhecida como camada de tradução, pois ela é responsável por converter o dado recebido pela camada de aplicação em um formato comum que pode ser utilizado pela pilha de protocolos. Por exemplo, se o programa estiver utilizando um código de página diferente do ASCII (*American Standard Code for Information Interchange*), a camada de apresentação será responsável por traduzir o dado recebido para o padrão ASCII. Esta camada também pode ser utilizada para comprimir e/ou criptografar os dados. Se a criptografia for utilizada, os dados vão trafegar entre camadas 1 e 5 criptografados, e só serão descriptografados na camada de apresentação do computador de destino.

1.1.7 Camada de Aplicação

De acordo com Torres (2007, p.2) "a camada de aplicação faz a interface entre o programa que está enviado ou recebendo dados e a pilha de protocolos". Segundo Tanenbaum (2011) essa camada possui uma série de protocolos comumente necessários para os usuários. Dentre os diversos protocolos de aplicação, de longe o mais utilizado é o HTTP (*HyperText Transfer Protocol*), que constitui a base da *World Wide Web*. Quando um navegador deseja acessar uma página *Web*, ele envia o nome da página ao servidor que hospeda esta página, utilizando o protocolo HTTP. O servidor, então, transmite a página ao navegador.

1.2 MODELO TCP/IP

O modelo de referência mais conhecido (e mais antigo) é o TCP/IP. Este modelo surgiu da rede ARPANET, que foi uma rede de pesquisa patrocinada pelo Departamento de Defesa dos Estados Unidos visando à conexão de inúmeras redes (DANTAS, 2002). No começo centenas de universidades e repartições públicas foram sendo conectadas, utilizando linhas telefônicas dedicadas. Mais tarde, quando surgiram as redes de rádio e satélite, os protocolos existentes começaram a ter problemas de interligação com elas, o que forçou a criação de uma

nova arquitetura de referência. Desse modo, é possível dizer que a capacidade para conectar várias redes de maneira uniforme foi um dos principais objetivos do projeto. A arquitetura concebida ficou conhecida como modelo de referência TCP/IP, por causa de seus dois principais protocolos (TANENBAUM, 2011).

Conforme Costa (2005) a pilha de protocolos TCP/IP é formada por níveis ou camadas lógicas. Cada protocolo de comunicação empregado na Internet é um software que interage com outros softwares (protocolos) na mesma máquina e em máquina remotas. Com a divisão em camadas, problemas específicos de comunicação em rede podem ser mais facilmente tratados, delimitando a atuação dos protocolos a áreas especificas.

Assim como no modelo OSI, no modelo TCP/IP cada protocolo comunica-se diretamente, na mesma máquina, com seus protocolos de nível lógico imediatamente abaixo e acima dele. Nessa comunicação, dados são coletados de um nível lógico, processados pelo nível intermediário e transferidos para o nível lógico seguinte. Da mesma forma, os protocolos comunicam-se com seus pares correspondentes em máquinas remotas, utilizando, para tanto, a infraestrutura de comunicação (COSTA, 2005, p.2-3).

O modelo de referência TCP/IP é constituído por cinco camadas. As quatro primeiras camadas fornecem padrões físicos, interface de rede, interconexão em rede e funções de transporte que correspondem às quatro primeiras camadas do modelo OSI. Entretanto, as três camadas superiores do modelo OSI são representadas no TCP/IP por uma única, chamada de camada de aplicação (FOROUZAN e FEGAN, 2009).

Como foi dito o modelo TCP/IP possui cinco camadas, mas geralmente a camada física não é referenciada entre as camadas, muitos autores, como Tanenbaum, apenas apresentam o modelo TCP/IP com quatro camadas. A camada física do modelo TCP/IP tem a mesma função da camada física do modelo OSI.

1.2.1 Camada de Interface de Rede

A camada de interface de rede é o responsável pelo envio dos datagramas de um computador qualquer para outro computador, independentemente de sua localização (DANTAS, 2002).

Os datagramas IP, provenientes do nível de rede, são encapsulados em quadros de nível de enlace e transmitidos no meio físico de acordo com os procedimentos da tecnologia de enlace adotada. Porém, os datagramas oriundos desse nível lógico são desencapsulados e encaminhados ao nível de rede. Numa comunicação tradicional, esse procedimento pode ser repetido várias vezes, uma vez que os *datagramas* IP podem atravessar diversos enlaces logicamente e tecnologicamente heterogêneos (COSTA, 2005).

1.2.2 Camada de Rede

Os serviços de rede oferecidos pela Internet são disponibilizados pelo protocolo IP. Segundo Torres (2007, p.4) "A camada de rede é responsável por adicionar um cabeçalho ao pacote de dados recebidos da camada de transporte onde, entre outros dados de controle, será adicionado também o endereço IP de origem e o endereço IP de destino". Isto é, o endereço IP do computador que está enviando os dados e o endereço IP do computador que vai recebê-los.

O protocolo IP também é responsável pelo roteamento de pacotes, que nada mais é que a escolha de um caminho por onde o pacote será enviado (COSTA, 2005). Quando são solicitados dados de um servidor de Internet, por exemplo, estes dados passam por vários locais, chamados de roteadores, antes de chegar a seu computador. Cada roteador no meio do caminho é também conhecido como "salto" (TORRES, 2007).

1.2.3 Camada de Transporte

A finalidade dessa camada é permitir que hosts de origem e destino possam manter uma conversação, exatamente como ocorre na camada de transporte do modelo OSI (TANENBAUM, 2011).

Quando uma aplicação deseja utilizar os recursos de comunicação com a Internet, os pontos de acesso para esse serviço são os protocolos de transporte. Os dados transmitidos da camada de aplicação para a camada de transporte são chamados de dados de usuário, estes dados possuem apenas significado para o emissor e o receptor dos datagramas IP. No protocolo IP este comportamento é diferente, onde os datagramas em nível de rede são processados por cada nó roteador no caminho entre origem e destino. Por ter significado apenas para a origem e destino da comunicação, costuma-se dizer que a comunicação é fim-a-fim (COSTA, 2005).

Conforme Costa (2005) o protocolo IP oferece um serviço de datagramas não confiável, ou seja, não existe garantia na entrega dos dados transmitidos. Os dados transmitidos podem chegar desordenados (se comparado com outros dados transmitidos do mesmo nó), também podem chegar duplicados ou serem perdidos. Dessa forma, algum mecanismo deve ser fornecido para garantir a confiabilidade na transmissão dos dados, caso este serviço seja necessário por determinada aplicação. Para oferecer a confiabilidade necessária na comunicação em rede, os protocolos de transporte podem ser utilizados.

Dentre os diversos mecanismos para garantir a confiabilidade dos dados, o mais comum é oferecer operações de notificação de erro, perda ou duplicação de pacotes. Caso um pacote IP seja perdido, devido a algum problema na rede, a informação não estará mais disponível ao nível de transporte. Através de mensagens de controle dos protocolos de transporte, é possível informar a falta das informações necessárias ao destino dos dados, requisitando, assim, uma ação de retransmissão dos dados. Mas se essas transmissões são feitas e de que modo, isso varia de acordo com o protocolo de transporte utilizado (COSTA, 2005).

Um conceito importante para a camada de transporte é o de porta de protocolo. Uma porta é o ponto de acesso do nível de transporte ao nível de aplicação. Seu valor, um inteiro positivo, indica a camada de transporte a que aplicação determinado dado recebido deve ser entregue (COSTA, 2005).

Para iniciar uma comunicação, se utiliza portas padronizadas para cada aplicação, as quais possuem valores entre 0 e 1023, elas são conhecidas como *Well Know Ports* (Portas Bem Conhecidas). Sem a padronização das portas não seria possível alcançar o protocolo de aplicação desejado, uma vez que o valor da porta na estação remota, poderia ser qualquer um.

1.2.4 Camada de Aplicação

O modelo TCP/IP não possui as camadas de sessão ou de apresentação. Não foi percebida qualquer necessidade para elas. Ao invés disso, as próprias aplicações simplesmente incluem quaisquer funções das camadas de sessão e apresentação que forem necessárias. A experiência com o modelo OSI demonstrou que essa visão está correta, pois ambas as camadas são pouco utilizadas na maioria das aplicações (TANENBAUM, 2011).

A camada de aplicação faz a comunicação entre os programas e os protocolos de transporte (TORRES, 2007). Este é o nível lógico mais alto na pilha TCP/IP, ela é responsável pela comunicação efetiva dos dados entre os usuários das estações (COSTA, 2005).

De acordo com Costa (2005, p.8) "cada aplicação possui requisitos próprios de operação, a escolha do protocolo de transporte a ser utilizado depende do levantamento dessas necessidades".

O quadro 1.1 apresenta os protocolos mais conhecidos na camada de aplicação. A cada protocolo está associada uma porta de protocolo única e bem conhecida.

Quadro 1.1 - Alguns protocolos de aplicação e suas portas associadas

Protocolo de Aplicação	Porta
FTP (File Transfer Protocol)	20/21
Telnet (Terminal Remoto)	23
SMTP (Simple Mail Transfer Protocol)	25
DNS (Domain Name System)	53
HTTP (Hyper Text Transfer Protocol)	80
SNMP (Simple Network Management Protocol)	161

Fonte: Desenvolvido pelo autor, adaptado de Costa (2005, p.8)

O uso de porta permite ao protocolo de transporte saber qual tipo de conteúdo do pacote de dados ela pertence, necessário aos procedimentos de multiplexação oferecidos pela camada de transporte.

2 PROTOCOLOS DE TRANSPORTE

Conforme Dantas (2002) os protocolos podem ser entendidos como um conjunto de regras que determinam como deverá ocorrer a comunicação entre duas estações em uma rede de computadores e como os erros devem ser detectados e tratados. O principal objetivo dos protocolos de transporte é oferecer um serviço confiável, eficiente e econômico a seus usuários.

Segundo Costa (2005) a Internet originalmente definiu os protocolos UDP e TCP como padrões para transmissão dados. As aplicações que desejam uma comunicação rápida e flexível recorrem ao protocolo UDP, enquanto o TCP é destinado às comunicações confiáveis. Entretanto, algumas aplicações precisam de confiabilidade, porém, são limitadas ou tornam-se menos eficientes devido à operação do TCP.

Com a evolução das aplicações e da Internet, e a necessidade de obter melhores resultados em determinadas situações, surgiu à necessidade de utilização de novos protocolos, como o SCTP. Segundo Stewart et al. (2000, tradução nossa) o protocolo SCTP é um protocolo de transporte confiável operando no topo de uma rede de pacotes sem conexão como IP.

Os protocolos UDP e TCP serão abordados nas próximas seções. Já o protocolo SCTP será detalhado de forma mais abrangente no próximo capítulo.

2.1 PROTOCOLO UDP

O protocolo UDP é um padrão TCP/IP definido na RFC 768. O UDP é utilizado por algumas aplicações por oferecer transporte rápido, leve e não confiável de dados entre hosts TCP/IP. Devido a sua simplicidade, a utilização do protocolo UDP é ideal para aplicações que não necessitem de muito controle por parte do nível de transporte (COSTA, 2005).

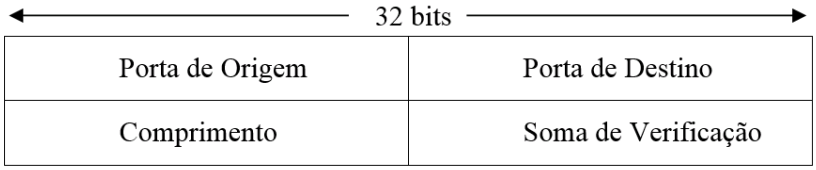
De acordo com Costa (2005, p12) “as mensagens UDP são chamadas de datagramas UDP, uma vez que possuem os mesmos conceitos dos datagramas IP: sem conexão e não confiável”. O UDP oferece um meio para as aplicações enviarem datagramas IP encapsulados sem que seja necessário estabelecer uma conexão (TANENBAUM, 2011).

O protocolo UDP não realiza controle de fluxo, controle de congestionamento ou retransmissão após a recepção de um segmento incorreto (TANENBAUM, 2011). “Como as

mensagens UDP podem ser perdidas, duplicadas ou corrompidas, já que são transmitidas por serviços do protocolo IP, é de se esperar que a confiabilidade numa comunicação usuária UDP seja garantida por protocolos no nível de aplicação" (COSTA, 2005, p.12).

A figura 2.1 apresenta o formato do datagrama UDP. O cabeçalho é muito simples, ele apresenta quatro campos de dois octetos cada. Os dois primeiros campos PORTA DE ORIGEM e PORTA DE DESTINO de acordo com Tanenbaum (2011, p.340) "[..]servem para identificar os pontos extremos nas máquinas de origem e destino[..]. Pense nas portas como caixas de correio que as aplicações podem utilizar para receber as correspondências".

Figura 2.1 - Cabeçalho do protocolo UDP



Fonte: Desenvolvido pelo autor, adaptado de Tanenbaum (2011, p.340)

> [...] o principal valor de ter o UDP em relação ao uso do IP bruto é a adição das portas de origem e destino. Sem os campos portas, a camada de transporte não saberia o que fazer com o pacote que chega. Com eles, a camada entrega o segmento encapsulado à aplicação correta (TANENBAUM, 2011, p.340).

O terceiro campo indica o COMPRIMENTO, que descreve quantos bytes terá o pacote completo, o que inclui o cabeçalho de 8 bytes e os dados. O comprimento mínimo é de 8 bytes, que é o comprimento do cabeçalho, considerando um pacote sem dados. O comprimento máximo é de 65.515 bytes, que o maior número que caberá em 16 bits, devido ao limite de tamanho dos pacotes IP (TANENBAUM, 2011).

O quarto campo compreende o SOMA DE VERIFICAÇÃO, sua utilização é opcional, devendo ser marcado com zeros caso sua utilização não seja necessária. Mas este campo pode ser utilizado para a detecção de erros na transmissão do datagrama UDP. A partir das informações presentes no datagrama UDP, em adição do pseudocabeçalho oriundo do protocolo IP, uma soma de verificação é calculada empregando-se algoritmos de soma de complementos. O pseudocabeçalho utilizado neste algoritmo não é transmitido nem adicionado ao cálculo do comprimento do datagrama (COSTA, 2005).

Quando o datagrama chega ao seu destino de comunicação, a soma de verificação é recalculada e comparada ao valor presente no cabeçalho, a diferença entre os valores indica a ocorrência de erros durante a transmissão do datagrama pela rede. Caso um erro seja encontrado, o datagrama UDP é simplesmente descartado. Nenhum mecanismo de correção de erro ou retransmissão de mensagens corrompidas, duplicadas ou perdidas é oferecido pelo UDP, devendo esses procedimentos ser esperados pelos protocolos de nível mais alto (COSTA, 2005).

2.2 PROTOCOLO TCP

O protocolo TCP é um padrão TCP/IP definido na RFC 793. Segundo a RFC 793 (1981) o protocolo TCP fornece um serviço de entrega de pacotes confiável e orientado por conexão. Ser orientado por conexão significa que todos os aplicativos baseados em TCP como protocolo de transporte, antes de iniciar a troca de dados, precisam estabelecer uma conexão.

Segundo Costa (2005) o protocolo TCP foi projetado para oferecer confiabilidade e controle não existente na camada de rede da Internet. Conforme Tanenbaum (2011) uma rede interligada é diferente de uma única rede, pois elas geralmente apresentam topologias, larguras de banda, atrasos, tamanhos de pacotes e outros parâmetros totalmente diferentes. O TCP foi desenvolvido para se adaptar dinamicamente a todas essas propriedades das redes interligadas.

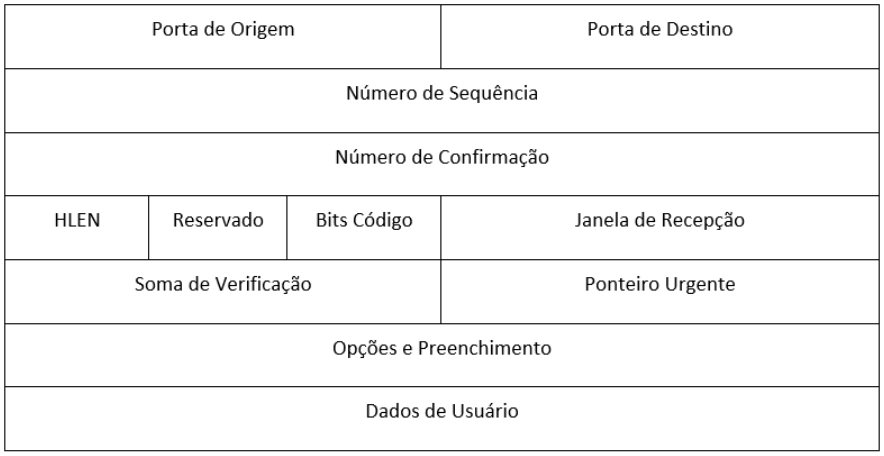
Em uma transmissão de dados, os pacotes podem ser perdidos ou descartados, pois o protocolo IP não fornece nenhum mecanismo de controle de erros. Utilizando o protocolo TCP, a camada de aplicação pode se livrar da tarefa de controle, necessária quando o protocolo UDP é empregado, facilitando o processo de desenvolvimento de softwares que necessitem de transmissão confiável de dados (COSTA, 2005).

2.2.1 Estabelecimento da conexão TCP

A comunicação do protocolo TCP é realizada através de uma conexão *full-duplex. Full-duplex* quer dizer que o tráfego pode ser feito em ambas as direções ao mesmo tempo. Como o TCP é um protocolo orientado a conexão, ele utiliza os endereços IP das estações e as portas de transporte dos pares da comunicação para identificar as conexões, garantindo que determinada conexão tenha uma identificação única na rede, em determinado momento (COSTA, 2005).

A figura 2.2 apresenta o segmento de um pacote TCP. Conforme Tanenbaum (2011) cada segmento começa com um cabeçalho de formato fixo de 20 bytes. O cabeçalho fixo pode ser seguido por opções de cabeçalho. Depois das opções, pode haver até 65.535 - 20 - 20 = 65.495 bytes de dados, sendo que o primeiro valor 20 se refere ao cabeçalho IP e o segundo ao cabeçalho TCP. Segmentos sem nenhum dado são válidos e são normalmente utilizados para confirmações e mensagens de controle.

Figura 2.2 - Segmento de um pacote TCP



Fonte: Costa (2005, p.17)

Os campos PORTA DE ORIGEM e PORTA DE DESTINO, assim como no protocolo UDP, identificam os pontos terminais da conexão. A porta TCP mais o endereço IP do seu host formam uma extremidade exclusiva de 48 bits. As extremidades de origem e destino juntas identificam a conexão (TANENBAUM, 2011).

O campo NÚMERO DE SEQUÊNCIA possui 32 bits, ele identifica a posição do segmento no fluxo de transmissão dos segmentos enviados. O campo NÚMERO DE CONFIRMAÇÃO também possui 32 bits, ele descreve qual deve ser o número de sequência do próximo segmento que uma estação espera receber (RFC 793, 1981).

O campo HLEN informa quantas palavras de 32 bits existem no cabeçalho TCP. Essa informação é necessária, porque o campo OPÇÕES tem tamanho variável, assim como acontece com o cabeçalho (TANENBAUM, 2011).

O campo RESERVADO tem 4 bits, e assim como seu nome indica, está reservado para uso futuro, não sendo utilizado no momento. Segundo Tanenbaum (2011, p.350) "o fato de esse campo ter sobrevivido intacto por 30 anos (pois apenas 2 dos 6 bits reservados originais foram reivindicados) é a prova de como o TCP é bem organizado. Protocolos menores teriam precisado dele para corrigir bugs no projeto original."

O campo SOMA DE VERIFICAÇÃO, assim como no UDP, é oferecido para aumentar a confiabilidade na entrega. "Ele confere o *checksum* do cabeçalho, dos dados e do pseudocabeçalho exatamente da mesma maneira que o UDP, exceto que o pseudocabeçalho tem o número de protocolo para o TCP e o *checksum* é obrigatório" (TANENBAUM, 2001, p.351).

O campo OPÇÕES, de acordo com Tanenbaum (2011), foi projetado para oferecer recursos extras, ou seja, recursos que não foram previsto no cabeçalho comum. Muitas opções foram definidas e várias são utilizadas. Entre elas está a opção MSS (*Maximum Segment Size*), que permite cada host estipular o tamanho máximo do segmento. A opção SACK (*Selective ACKnowledgement*) que permite um receptor informar a um transmissor os intervalos de números de sequência que ele recebeu.

O campo CÓDIGOS DE BITS é composto por seis 6 bits, cada um com significado próprio. O quadro 2.1 apresenta os bits e o significado deles quando marcados.

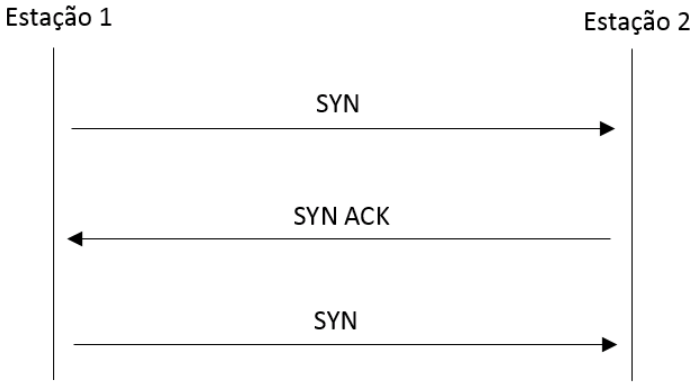
Quadro 2.1 - Códigos de bit válidos

Bit de Controle	**Significado do Bit**
URG	Segmento contém dados urgentes
ACK	Segmento também está fazendo reconhecimento
PSH	Operação PUSH deve ser utilizada
RST	Restabelecer a conexão
SYN	Sincronização inicial da conexão
FIN	Encerramento da conexão

Fonte: Costa (2005, p.19).

Segundo Costa (2005) para estabelecer uma conexão TCP corretamente, é utilizado um processo de troca de três segmentos, contendo, respectivamente, os bits SYN, SYN ACK e ACK marcados. O estabelecimento de conexões TCP é definido como um *handshake* de três vias (*three way handshake*). A figura 2.3 apresenta a troca de mensagens para o estabelecimento de uma conexão TCP.

Figura 2.3 - Estabelecimento conexão TCP



Fonte: Costa (2005, p.19)

2.2.2 Controle de Erro

O controle de erro oferecido pelo TCP, conforme Costa (2005, p.20) "garante que segmentos perdidos ou descartados sejam recuperados por mecanismos de retransmissão e que segmentos duplicados sejam devidamente tratados".

O TCP utiliza uma técnica que visa oferecer confirmação positiva dos segmentos enviados, indicando que os segmentos não reconhecidos foram perdidos ou descartados. Na transmissão dos dados, o receptor informa continuamente, ao transmissor, os segmentos que foram recebidos. A confirmação é feita através do campo NÚMERO DE RECONHECIMENTO que envia o bit ACK ao transmissor para cada segmento recebido. Quando o transmissor recebe um segmento com o bit ACK marcado, ele é capaz de avaliar se os segmentos enviados foram corretamente recebidos ou se algum procedimento de retransmissão é necessário (COSTA, 2005).

2.2.3 Controle de Fluxo

O controle de fluxo do TCP é administrado por meio de uma janela deslizante. Segundo Pinheiro (2010) a janela deslizante permite que, o remetente transmita mais que um segmento de dados antes de receber uma confirmação (mensagem ACK), múltiplos segmentos podem ser enviados antes de receber qualquer mensagem de reconhecimento. Este procedimento garante maior eficiência à comunicação.

> Este controle de fluxo é feito através do campo JANELA DESLIZANTE, de dois octetos. Esse campo informa o valor atual do *buffer* de recepção, em octetos, do emissor do segmento contendo o campo JANELA DESLIZANTE considerado. Caso o receptor esteja impossibilitado de receber mais segmentos, devido a seu *buffer* de recepção estar completamente ocupado ou se houver alguma sobrecarga de processamento, essa condição pode ser informada com a indicação de uma janela deslizante com valor zero (COSTA, 2005, p.21)

Segundo Costa (2005) este campo está presente em todos os segmentos, ou seja, as informações do tamanho atual do *buffer* de recepção são indicadas continuamente, o que garante a dinâmica necessária aos procedimentos de controle de fluxo.

2.2.4 Controle de Congestionamento

Quando a carga oferecida a uma rede é maior que sua capacidade, acontece um congestionamento. De acordo com Tanenbaum (2011, p.359),

> [...] a camada de rede detecta o congestionamento quando as filas se tornam grandes nos roteadores e tenta gerenciá-lo, mesmo que apenas descartando pacotes. Cabe a camada de transporte receber o *feedback* do congestionamento da camada de rede e diminuir a velocidade do tráfego que está enviando para a rede.

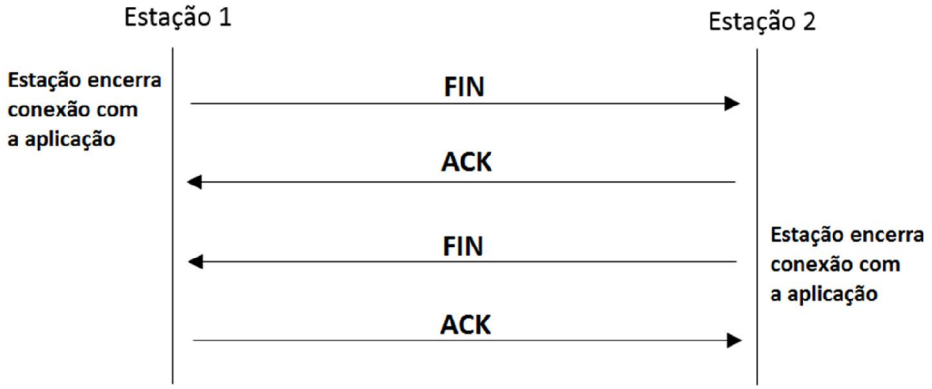
O protocolo TCP oferece diversos algoritmos para o controle de congestionamento, além de diminuir a taxa de transmissão em redes congestionadas, o TCP tem um mecanismo que aumenta gradativamente a taxa de transmissão, que pode ser utilizada para não provocar novos congestionamentos quando a rede começa a se recuperar da sobrecarga anterior (COSTA, 2005).

2.2.5 Encerramento da conexão TCP

Para encerrar uma conexão TCP, um dos lados precisa enviar um segmento com o bit FIN marcado, o que significa que não há mais dados a serem transmitidos. Se o bit FIN for

confirmado, este lado será desativado para novos dados. No entanto, os dados podem continuar a fluir indefinidamente no outro sentido, este processo é conhecido como *half closed* (meio-fechado). Somente quando os dois sentidos da conexão estiverem desativados, a conexão será encerrada. De maneira geral, são necessários quatro segmentos TCP para encerrar uma conexão, um FIN e um ACK em cada sentido. Porém, é possível que o primeiro ACK e o segundo FIN ocupem o mesmo segmento (TANENBAUM, 2011). A figura 2.4 apresenta a troca de mensagens para encerrar uma conexão TCP.

Figura 2.4 - Encerrando conexão TCP



Fonte: Costa (2005, p.22)

3 PROTOCOLO SCTP

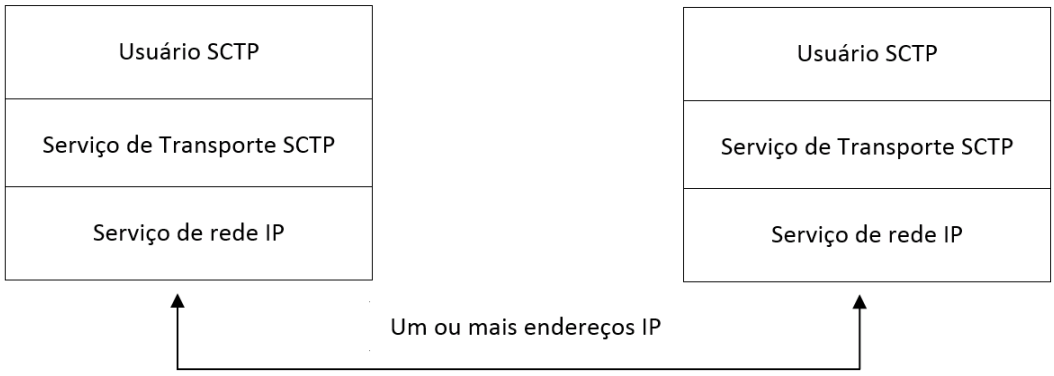
O SCTP é um protocolo de transporte relativamente novo, ele surgiu do trabalho dos pesquisadores Randall Stewart e Qiaobing Xie, que contaram com o apoio do comitê SIGTRAN (*SIGnalling TRANsport*) da IETF (Internet *Engineering Task Force*). O objetivo do comitê era encontrar ou criar um protocolo adequado para a transmissão sobre o TCP/IP de mensagens de sinalização telefônica. Protocolos como o TCP e UDP foram julgados inadequados para essa tarefa, o que levou a criação do SCTP (PFÜTZENREUTER, 2007).

Inicialmente, o SCTP foi utilizado apenas para a troca de mensagens telefônicas na Internet, mas conforme Costa (2005) as características do SCTP fazem com que seu uso seja interessante em aplicações comuns da Internet, o que resultou em sua incorporação ao grupo dos tradicionais protocolos de transporte da pilha TCP/IP.

> A grande vantagem do SCTP é que ele fornece um número de funções robustas e flexíveis a diversas aplicações, funções essas não presentes nos protocolos UDP ou TCP. Além disso, o SCTP também possibilita benefícios às aplicações que necessitam de performance adicional (COSTA, 2005, p.24).

A definição formal do protocolo está contida na RFC 2960 (STEWART et al, 2000). Desde a sua definição em outubro de 2000, várias correções e melhorias já foram incorporadas a RFC original. A RFC 4460 (STEWART et al, 2006) contém uma listagem detalhada de todos os problemas encontrados durante os seis anos de experiência com a implementação e testes do SCTP, juntamente com as correções sugeridas. A figura 3.1 apresenta a localização lógica do SCTP na pilha TCP/IP.

Figura 3.1 - Localização lógica do SCTP na pilha TCP/IP



Fonte: Desenvolvido pelo autor, adaptado de Costa (2005, p.23)

3.1 CARACTERÍSTICAS DO SCTP

Segundo Costa (2005) o SCTP apresenta todas as características de um protocolo de transporte confiável, como o TCP, porém, possui funcionalidades adicionais em relação a este último, conhecidas como múltiplos fluxos (*multistreaming)* e múltiplos endereços (*multihoming)*. De acordo com Stewart et al (2001) o SCTP apresenta algumas características herdadas do TCP, mas oferece outras funcionalidades adicionais como:

• Entrega confirmada, livre de erros e não duplicada de dados de usuário;

• Fragmentação de dados em conformidade com o MTU (*Maximum Transmission Unit*) descoberto do caminho;

• SCTP suporta vários modos de entrega incluindo ordem de transmissão estrita (como o TCP), parcialmente ordenado e entrega desordenada (como o UDP).

• Entrega sequencial de mensagens de usuário em múltiplos fluxos, com opção para entrega por ordem de chegada de mensagens de usuário individuais;

• Empacotamento opcional de múltiplas mensagens de usuário em um único pacote SCTP;

• Tolerância a falhas de rede através do suporte a multicaminhos, SCTP envia pacotes para um endereço IP de destino, mas pode redirecionar mensagens para um suplente se o endereço IP atual torna-se inacessível.

Da mesma maneira que o TCP, o SCTP fornece um serviço confiável de transporte, garantindo que os dados sejam entregues sem erro e não os duplicando e sempre em sequência. O SCTP consegue descobrir quando os dados são descartados, reordenados, duplicados ou corrompidos, retransmitindo-os quando necessário (COSTA, 2005).

Segundo Costa (2005, p25),

> A confiabilidade de transmissão advém do fato de o SCTP ser orientado a conexão. Uma conexão necessita permanecer ativa durante toda a comunicação, devendo ser explicitamente iniciada antes do envio de qualquer dado de usuário. Este mesmo procedimento ocorre com o protocolo TCP. Contudo, embora o SCTP seja orientado a conexão, como o TCP, a conexão SCTP é definida como uma *associação* que possui um contexto e significado mais amplo que a conexão TCP.

Assim como o TCP, o SCTP é *rate adaptive,* o que garante adaptação dinâmica às variações e problemas ocorridos na rede, como congestionamentos. De acordo com Costa (2006, p.25) "muitos dos algoritmos de controle de congestionamento utilizados pelo TCP foram reaproveitados ou reformulados para o SCTP, uma vez que suas funcionalidades já estavam largamente testadas em cenários reais de congestionamento".

3.2 ORIENTAÇÃO A MENSAGENS

Apesar dos protocolos TCP e SCTP possuírem muitas semelhanças, existem muitas abordagens diferentes. Uma delas diz respeito a como os dados de usuário são tratados pelo protocolo de transporte. O protocolo SCTP é orientado a mensagens, já o protocolo TCP é orientado a octetos. Sendo assim, no SCTP os dados de usuário a serem transmitidos são tratados como blocos independentes, com significado próprio. O TCP não preserva qualquer estrutura dos dados dentro do fluxo de transmissão (COSTA, 2005).

Segundo Forouzan (2008), um segmento TCP pode transportar simultaneamente dados e informações de controle. Desta maneira, todos os dados de usuário precisam receber algum tipo de marcação especial no nível de aplicação. Para o TCP, os dados são apenas octetos, e podem ser transmitidos de qualquer forma, aos poucos, misturados, com outros octetos. Se as aplicações não colocarem algum tipo de marcação, a aplicação que recebesse os dados não saberia identificar a mensagem recebida uma vez que diversas informações poderiam ter sido misturadas.

A orientação a mensagens do SCTP facilita o desenvolvimento de aplicações que precisam de confiabilidade ao nível de transporte. As mensagens de texto e controle são enviadas pelo SCTP como mensagens de texto e de controle, pois o SCTP não mistura as informações de usuário com significados diferentes. Desta maneira, não são necessárias marcações ao nível de aplicação (COSTA, 2005).

3.3 ASSOCIAÇÃO SCTP

Associação é o nome dado as conexões SCTP, uma associação, como uma conexão TCP, possui caráter lógico fim-a-fim, sendo controlada por variáveis e protocolos da camada de transporte. Segundo Costa (2005) a principal diferença entre uma associação SCTP e uma

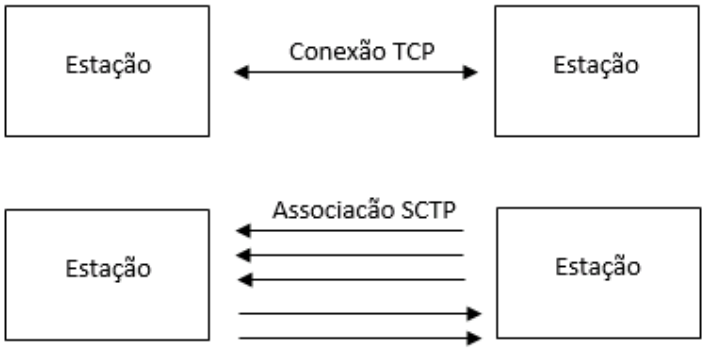
conexão TCP é que a primeira pode ter um número arbitrário de fluxos *simplex* (unidirecionais) definidos durante o início da associação, enquanto a última é formada apenas por um único fluxo *full-duplex*. A ideia de múltiplos fluxos é uma característica marcante do SCTP, que oferece uma maior eficiência na comunicação.

Cada fluxo SCTP é um canal de comunicação unidirecional, é perfeitamente possível haver um número desigual de fluxos em cada direção. Uma associação SCTP que pretenda simplesmente emular TCP usa dois fluxos, um em cada direção (PFÜTZENREUTER, 2004, p.27).

O SCTP não suporta o estado "meio aberto" como o TCP, onde um dos lados da comunicação pode continuar a enviar dados mesmo que seu par tenha "deixado" a conexão. Quando uma associação é encerrada por qualquer um dos lados, os terminais comunicantes param de aceitar novos dados da camada de aplicação (STEWART et al, 2000). "Dessa forma, o SCTP evita os problemas enfrentados pelo TCP com as conexões meio abertas" (COSTA, 2005, p.27).

A figura 3.2, apresenta um modelo conceitual de múltiplos fluxos das associações SCTP em comparação às conexões *full-duplex* do TCP.

Figura 3.2 - Modelo Conceitual das Associações SCTP



Fonte: Desenvolvido pelo autor, adaptado de Costa (2005, p.28).

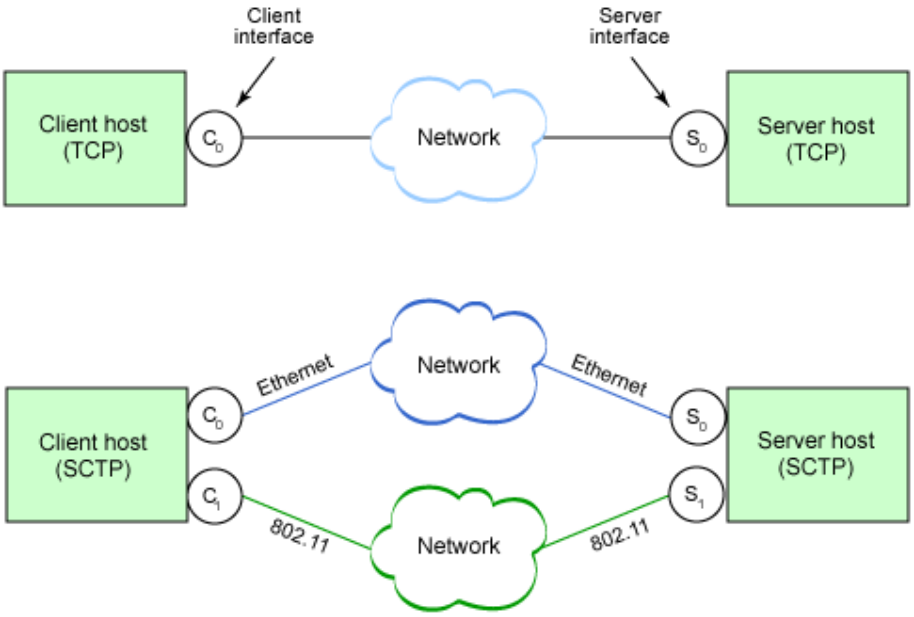
3.4 MÚLTIPLOS CAMINHOS

"A maior diferença entre o TCP e o SCTP é o *multihoming* (múltiplos caminhos), o SCTP foi projetado para estabelecer uma comunicação robusta entre dois pontos, sendo que

cada um pode ser acessado por mais de um endereço de transporte" (STEWART et al, 2000, tradução nossa).

Em uma conexão TCP pode haver apenas um endereço IP de origem e um endereço IP de destino. Mesmo que o remetente ou receptor seja um *host multihomed* (conectado a mais de um endereço físico com vários endereços IP), apenas um desses IP por extremidade pode ser utilizado durante a conexão. Uma associação SCTP, suporta o serviço *multihoming* (múltiplos caminhos), isso possibilita o *host* remetente e o *host* receptor definirem vários endereços IP em cada extremidade de uma associação (FOROUZAN e FEGAN, 2009). A figura 3.3 ilustra a diferença entre uma conexão TCP e uma associação SCTP.

Figura 3.3 - Conexão TCP vs Associação SCTP



Fonte: Jones (2006, p.1)

A falha de rede em uma conexão TCP pode causar indisponibilidade de comunicação entre os terminais comunicantes. Com o uso de mais de um endereço IP numa associação SCTP, acessos redundantes podem ser utilizados para reforçar a comunicação.

"Até o momento o SCTP não é capaz de realizar balanceamento de carga entre os caminhos redundantes. A função de *multihoming* é utilizada apenas para fornecer redundância em caso de problemas na rede" (AMER, 2013, tradução nossa).

Na associação SCTP, existem um par de endereços primários (um para cada estação) e um número qualquer de endereços alternativos. O SCTP monitora os caminhos da associação, se o caminho primário falhar, o protocolo envia o tráfego através do(s) caminho(s) alternativo(s), caso eles existam. Nem mesmo é necessário a aplicação ficar sabendo que ouve uma falha na rede (JONES, 2006). Segundo Costa (2005), os endereços alternativos são utilizados para aumentar a probabilidade de se alcançar o ponto remoto destino, em caso de falha do caminho primário. Se o problema persistir um novo endereço primário é estabelecido, através dos endereços alternativos possíveis e alcançáveis.

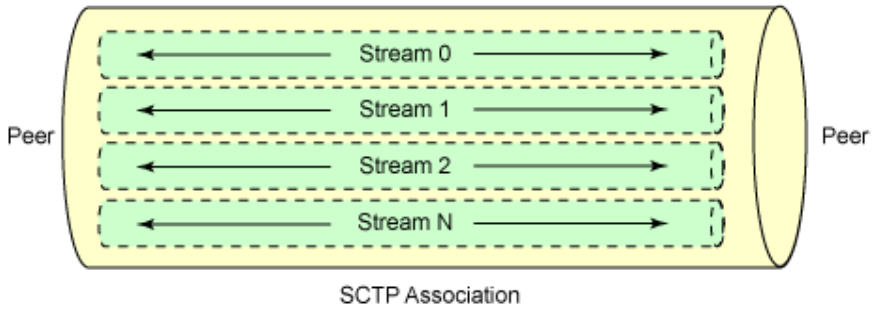
No início de uma associação acontece a troca de endereços primários e alternativos entre os terminais SCTP. Para todos esses endereços apenas uma única porta de transporte é utilizada por cada estação em determinada associação. Estações que informarem apenas um endereço IP (primário) não possuem endereços alternativos para a associação (COSTA, 2005).

3.5 MÚLTIPLOS FLUXOS

Uma conexão TCP entre dois terminais envolve um único fluxo *full-duplex*. O problema dessa estratégia é que uma perda em qualquer ponto no fluxo impede o envio do restante dos dados. Isso pode ser aceitável quando se está transferindo arquivos de texto, mas não quando são enviados dados em tempo real, como áudio ou vídeo.

Uma associação SCTP pode estabelecer um número arbitrário de fluxos *simplex* de transmissão e recepção (COSTA, 2006). Este conceito é conhecido como *multistreaming* (múltiplos fluxos). Se um dos fluxos for perdido, os outros ainda podem enviar seus dados. A ideia é semelhante a uma autoestrada com várias pistas. Por exemplo, uma pista pode ser utilizada para veículos normais e outra pista para veículos de transporte. Se o tráfego for bloqueado para os veículos normais, os veículos de transporte ainda podem chegar ao seu destino (FOROUZAN e FEGAN, 2009). A figura 3.4 ilustra o conceito de múltiplos fluxos:

Figura 3.4 - Múltiplos fluxos em uma associação SCTP



Fonte: Jones (2006)

"Todos os *streams* (fluxos) dentro da associação são independentes, mas relacionados com a associação" (JONES, 2006, tradução nossa).

Segundo Costa (2005) o número de fluxos de transmissão e recepção em uma associação pode ser definido pelo usuário dos serviços deste protocolo no início de uma associação. Como cada usuário informa o número de fluxos a ser suportado, basta apenas calcular um denominador comum entre os valores indicados para se definir a quantidade de fluxos em uma associação.

De acordo com Costa (2005) cada fluxo da associação recebe um identificador chamado de *Stream Identifier* (número de fluxo). Além deste identificador, cada mensagem recebe um identificador em relação à sua posição no fluxo. Esse identificador é chamado de *Stream Sequence Number* (número de sequência no fluxo). Estes dois identificadores formam o esquema de identificação de mensagens em relação ao escopo de ordenação. Por exemplo, uma determinada mensagem pode ter número de sequência no fluxo 5 e ter número de fluxo 2.

Como foi dito, a perda ou erro de um fluxo em particular não afeta os demais fluxos. Considere uma associação com dois fluxos, no fluxo número um, existem duas mensagens e o fluxo número dois possui três mensagens a serem transmitidas. Caso a mensagem de número de sequência dois, contida no fluxo número dois deva ser retransmitida, devido a erros na transmissão ou descarte por algum nó da rede, apenas o fluxo dois será afetado. Nenhuma mensagem contida no fluxo de número um será retransmitida ou transmitida com *delay* (atraso), os algoritmos de reordenação serão apenas aplicados no fluxo de número dois. O receptor pode continuar a enviar mensagens dos fluxos não atingidos para o nível de aplicação, enquanto

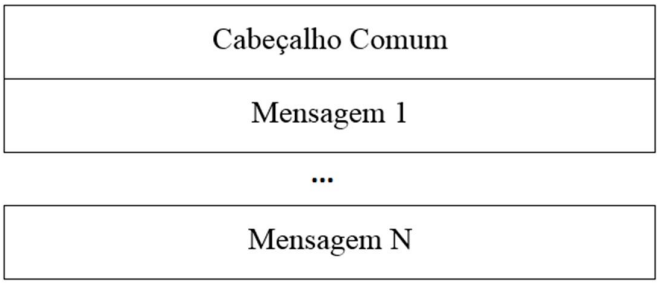
armazena, num *buffer,* as mensagens do fluxo afetado até que ocorra a retransmissão (COSTA, 2005).

Segundo Costa (2005) o SCTP utiliza um mecanismo de retransmissão seletiva, muito mais eficiente que o mecanismo de retransmissão padrão do TCP. No exemplo considerado, apesar da mensagem de número de sequência no fluxo dois ter sido perdida, a mensagem de número três não deve ser retransmitida. Apenas a mensagem com número de sequência no fluxo igual a dois será retransmitida. Sem a necessidade de retransmitir mensagens já recebidas, assim como a não interrupção da transmissão normal dos fluxos lógicos sem perda de mensagens, garante uma maior eficiência do SCTP frente ao TCP.

3.6 FORMATO DO PACOTE SCTP

O pacote SCTP é composto por um cabeçalho comum e uma ou mais mensagens. As mensagens podem conter informações de controle ou dados de usuário (STEWART et al, 2000). De acordo com Costa (2005, p.33) "para aumentar a eficiência de comunicação, múltiplas mensagens de usuário e mensagens de controle podem ser encapsuladas em um mesmo pacote SCTP, de acordo com a opção escolhida pelo usuário". O formato do pacote SCTP pode ser visto na figura 3.5.

Figura 3.5 - Formato do pacote SCTP



Fonte: Desenvolvido pelo autor, adaptado de Stewart (2000, p.15).

3.6.1 Cabeçalho Comum SCTP

O formato do cabeçalho comum SCTP, presente em todos os seus pacotes, é apresentado na figura 3.6. Segundo Costa (2005, p.34) "cada campo presente nesse cabeçalho, que somados

correspondem a um tamanho de doze octetos, é comum a todas as mensagens SCTP, de usuário ou de controle".

Figura 3.6 - Cabeçalho comum SCTP

Porta de Origem	Porta de Destino
Etiqueta de Identificação	
Soma de Verificação	

Fonte: Desenvolvido pelo autor, adaptado de Stewart et al (2000, p.16).

Os valores contidos nos campos PORTA DE ORIGEM e PORTA DE DESTINO são utilizados para multiplexação de comunicações diferentes. A multiplexação por porta também é utilizada pelos protocolos UDP e TCP, sendo esse um serviço básico e fundamental da camada de transporte. Os números de porta, juntamente com os endereços IP de origem e de destino, que podem ser mais de um por estação (*multihoming*), são utilizados para identificar as associações (COSTA, 2005).

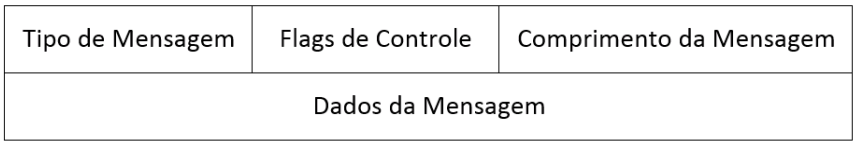
O campo ETIQUETA DE IDENTIFICAÇÃO, conforme Costa (2005) tem como finalidade verificar a autenticidade de determinado pacote, validando seu emissor. A maneira de realizar uma validação satisfatória é realizar uma troca de valores de identificação, que são gerados aleatoriamente pelas estações no estabelecimento da associação. Cada terminal calcula sua própria etiqueta, de forma aleatória, e a informa ao terminal par da associação. Desta forma é possível evitar que o valor deste campo seja previamente conhecido por uma das estações com intenções de transmitir mensagens forjadas para a associação.

O campo SOMA DE VERIFICAÇÃO, assim como no UDP e TCP é utilizado para detectar erros de transmissão, mas segundo Costa (2005) o SCTP utiliza um algoritmo mais robusto de soma de verificação. O algoritmo de soma de verificação do SCTP foi alterado na RFC 3309.

3.6.2 Formato das Mensagens SCTP

"Todo pacote SCTP contém uma ou mais mensagens, de usuário ou de controle. As mensagens são organizadas após o cabeçalho do pacote (COSTA, 2005, p.35)". A figura 3.7 apresenta o formato das mensagens SCTP.

Figura 3.7 - Formato das mensagens SCTP



Fonte: Desenvolvido pelo autor, adaptado de Costa (2005, p.36)

"O campo TIPO DE MENSAGEM identifica o tipo de informação contida neste campo. Seu valor varia de 0 a 254, o valor 255 é reservado para uso futuro como um campo de extensão" (STEWART et al, 2000, p.17, tradução nossa). O quadro 3.1 apresenta os valores para cada tipo de mensagem.

Quadro 3.1 - Tipos de Mensagens SCTP

Valor de Tipo	Mensagem correspondente
0	Dados de Usuário
1	Initiation (INIT)
2	Initiation Acknowledgement (INIT ACK)
3	Selective Acknowledgement (SACK)
4	Heartbeat Request (HEARTBEAT)
5	Heartbeat Acknowledgement (HEARTBEAT ACK)
6	Abort (ABORT)
7	Shutdown (SHUTDOWN)
8	Shutdown Acknowledgement (SHUTDOWN ACK)
9	Operation Error (ERROR)
10	State Cookie (COOKIE ECHO)
11	Cookie Acknowledgement (COOKIE ACK)
12	Reserved for Explicit Congestion Notification Echo (ECNE)
13	Reserved for Congestion Window Reduced (CWR)
14	Shutdown Complete (SHUTDOWN COMPLETE)
15 ao 62	Reservado
63	Extensão de Mensagem

64 ao 126	Reservado
127	Extensão de Mensagem
128 ao 190	Reservado
191	Extensão de Mensagem
192 ao 254	Reservado
255	Extensão de Mensagem

Fonte: Desenvolvido pelo autor, adaptado de Costa (2005, p. 36).

De acordo com Costa (2005) as mensagens que possuem valor igual a zero no campo TIPO DE MENSAGEM são mensagens de usuário, que, de fato, transmitem os dados na associação. As mensagens com valores de 1 a 14 são mensagens de controle necessárias às operações do SCTP. Para consultar de forma mais detalhada a função de cada mensagem de controle, é possível consultar a RFC 2960.

O campo FLAGS DE CONTROLE pode ser utilizado para oferecer opções de controle adicionais às mensagens, possui um octeto. A finalidade deste campo é utilizar seus bits como indicadores de alguma opção. A utilização deste campo é opcional (COSTA, 2005).

O campo COMPRIMENTO DA MENSAGEM indica o tamanho total da mensagem, incluindo os quatro octetos do cabeçalho padrão das mensagens. As mensagens podem ser de tamanhos variáveis (COSTA, 2005).

O campo DADOS DA MENSAGEM contém a informação real a ser transmitida. O uso e o formato deste campo dependem do campo TIPO DE MENSAGEM (STEWART et al. 2000, p.19).

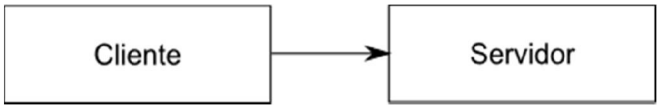
3.7 ESTADOS SCTP

Essa sessão descreve os estados de uma instância do SCTP, desde o estabelecimento de uma associação até seu término. O estabelecimento de uma associação SCTP é feito em um processo conhecido como *four-way handshake,* onde é necessário a troca de quatro mensagens entre as partes comunicantes. O lado passivo (que será chamado de servidor) não aloca recursos até que a terceira dessas mensagens tenha chegado e tenha sido validada. Esse mecanismo é necessário para garantir que a solicitação da criação de uma associação seja enviada de forma correta (CECHIN, 2008).

3.7.1 Estabelecimento da associação SCTP

Para demonstrar como é feito o estabelecimento da associação, será usada a nomenclatura de cliente-servidor da seguinte forma: o cliente é o terminal que deseja iniciar a associação; enquanto que o servidor é o terminal que aguarda a solicitação de um cliente para o estabelecimento da associação.

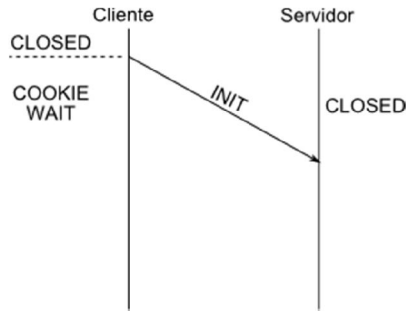
Figura 3.8 - Direção da associação



Fonte: CECHIN (2008)

No Cliente: quando a camada de aplicação deseja iniciar uma associação, ela preenche todas as estruturas de dados necessárias para montar uma mensagem INIT. Em seguida, essa mensagem (INIT) é enviada para o endereço de transporte de um servidor (ou seja, uma associação endereçada pela combinação de porta e endereço IP). Após o cliente ter enviado a mensagem INIT ele entra no estado COOKIE-WAIT (CECHIN, 2008).

Figura 3.9 - Inicialização de uma associação

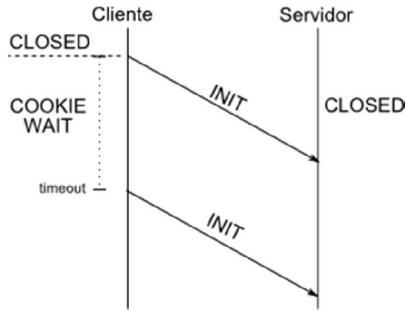


Fonte: CECHIN (2008)

Neste momento o cliente inicia um contador de tempo (temporizador). Caso o limite de tempo termine e nenhuma mensagem de resposta seja recebida, a mensagem não reconhecida é reenviada. Se uma mensagem chegar em tempo hábil, a contagem do temporizador é encerrada e nenhuma ação de retransmissão é tomada. O SCTP considera um limite máximo de retransmissões para mensagens perdidas, quando o limite é esgotado, o SCTP considera o caminho primário como não alcançável, devendo iniciar os procedimentos para utilização dos endereços alternativos (caso existam) ou encerrar a associação (COSTA, 2005).

A figura 3.10 demonstra o reenvio da mensagem INIT caso a primeira mensagem tenha sido perdida.

Figura 3.10 - Retransmissão da mensagem INIT

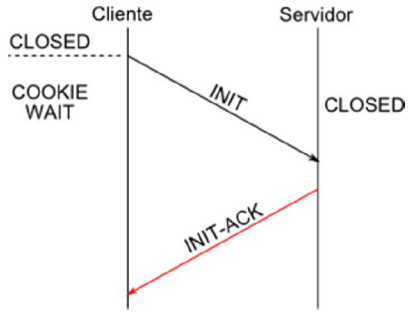


Fonte: CECHIN (2008)

No Servidor: quando o servidor recebe a mensagem INIT, ele inicia a geração de todas as variáveis necessárias para estabelecer uma associação. Em seguida é criado uma tabela *hash* segura para referenciar as variáveis com uma chave de segurança. Todos estes valores são colocados em uma estrutura de dados chamada COOKIE, juntamente com um valor de autenticação, conhecida como MAC (*Message Authentication Code*). O COOKIE criado no servidor é retornado para o cliente que solicitou a associação através da mensagem INIT ACK. O COOKIE é descartado pelo servidor que permanece no estado *closed*. Um temporizador pode ser utilizado para garantir que a mensagem enviada chegue ao seu destino (COSTA, 2005).

A figura 3.11 apresenta a resposta do servidor para a solicitação de uma nova associação.

Figura 3.11 - Resposta do servidor



Fonte: CECHIN (2008)

No cliente: quando o cliente, que permanece no estado COOKIE WAIT, recebe a mensagem INIT ACK do servidor, ele encerra o contador, monta uma mensagem COOKIE

ECHO, copia o COOKIE recebido na mensagem INIT ACK para a mensagem COOKIE ECHO e envia essa mensagem para o servidor. O cliente então inicia o contador do *cookie,* que é retransmitido sempre que o contador expirar, até que uma mensagem COOKIE ACK seja recebida do servidor. Depois de ter enviado o primeiro COOKIE ECHO, o cliente entra no estado *cookie echoed* (CECHIN, 2008). "Após o recebimento de uma mensagem COOKIE ACK do servidor, o cliente entra no estado *established,* no qual a associação, agora, está completamente definida (COSTA, 2005, p.61)".

No servidor: ao receber a mensagem COOKIE ECHO (que contém uma variável *cookie* como parâmetro), o servidor verifica o valor da variável MAC para certificar a validade da mensagem recebida. Essa verificação é feita para se assegurar que o *cookie* recebido é o mesmo que foi criado pelo servidor. Se o valor da variável MAC for igual ao valor esperado, o *cookie* presente na mensagem COOKIE ECHO é o mesmo *cookie* que o servidor havia criado (COSTA, 2005).

A partir dos valores contidos neste *cookie*, o servidor aloca recursos necessários para iniciar a associação SCTP. Em seguida o servidor entra no estado *established* e envia a mensagem COOKIE ACK para o cliente. A partir deste ponto, o servidor está pronto para receber e enviar mensagens de dados. A mensagem COOKIE ECHO já pode conter mensagens de dados, o servidor é quem decide se aceita ou descarta esses dados (CECHIN, 2008).

No cliente: quando o cliente receber a mensagem COOKIE ACK do servidor, ele aloca os recursos necessários para a associação e entra no estado *established.* Agora a associação está completamente definida.

Apenas para fins comparativos, é possível notar que o estabelecimento de uma conexão TCP é necessário a troca de três mensagens, enquanto que uma associação SCTP precisa trocar quatro mensagens. Pode parecer que o SCTP é menos eficiente do que o TCP, mas de acordo com Forouzan e Fegan (2009, p. 364) o SCTP permite a troca de dados no terceiro e no quarto pacotes, além de oferecer melhor segurança contra ataques de negação de serviço SYN, que são comuns no TCP. Ou seja, após a troca de duas mensagens, dados já pode ser transferidos.

3.7.2 Término da Associação SCTP

No SCTP, assim como no TCP, qualquer uma das partes envolvidas pode solicitar o término da associação. Ao contrário do TCP, o SCTP não permite uma associação "meia fechada". Se uma extremidade solicita o término da associação, a outra deve parar de enviar dados (FOROUZAN e FEGAN, 2009).

Segundo Costa (2005) existem dois tipos de abordagens no término de uma associação: a "coordenada", que garante a não perda de dados e a "forçada", que não garante a entrega dos dados que estão sendo transmitidos ou dos dados que estão mantidos em *buffers* da associação.

O término coordenado é baseado nas mensagens SHUTDOWN. Quando um dos terminais (cliente ou servidor) deseja encerrar a comunicação, ele para de receber dados da camada de aplicação, e, após a confirmação dos dados já enviados, transmite a mensagem SHUTDOWN à outra ponta. Um temporizador é utilizado para retransmitir mensagens SHUTDOWN sem respostas (COSTA, 2005).

O receptor da mensagem SHUTDOWN responde com o envio da mensagem SHUTDOWN ACK assim que todos os dados presentes nos seus *buffers* tenham sido confirmados. Também é utilizado um temporizador para garantir o envio correto da mensagem (COSTA, 2005).

Quando a estação requisitante do término da associação receber a mensagem SHUTDOWN ACK, ela interrompe o temporizador e em seguida envia a mensagem SHUTDOWN COMPLETE a outra ponta. Todos os dados pertencentes a associação são excluídos, e a estação requisitante entra em estado *closed* (COSTA, 2005).

O receptor da mensagem SHUTDOWN COMPLETE também remove todos os dados pertencentes a associação e entra em estado *closed.* Agora de fato a associação está completamente encerrada pelas duas estações (COSTA, 2005).

Já o término forçado de uma associação pode ser solicitado por qualquer uma das extremidades, enviando a mensagem ABORT. O emissor desta mensagem libera todos os recursos alocados à associação após seu envio. O lado receptor não responde à mensagem ABORT, ele apenas encerra imediatamente a associação (COSTA, 2005).

4 BENCHMARKING

No âmbito da computação, *benchmarking* é o ato de comparação entre dois ou mais sistemas através de métricas. Este processo permite medir e avaliar o desempenho de um sistema, quando está realizando uma tarefa ou um conjunto de tarefas bem definidas.

As normatizações dos *Benchmarks* para avaliação de desempenho em redes de computadores devem estar baseadas nas metodologias definidas pela IETF. Estas metodologias são publicadas em documentos denominados RFC (*requests for comments*), que se referem a uma série de documentos que contém informações técnicas e organizacionais sobre a Internet. Eles apresentam muitos aspectos da rede de computadores, incluindo protocolos, procedimentos, programas e conceitos.

4.1 RFC 2544

A RFC 2544 descreve e define uma série de testes que podem ser utilizados para apresentar as características de desempenho de dispositivos de rede. Ela também descreve formatos específicos para reportar os resultados dos testes (BRADNER e MCQUAID, 1999).

Segundo Burgess (2004) a RFC 2544 define que os tamanhos dos quadros sejam normalizados, com tamanhos definidos em (64, 128, 256, 512, 1024, 1280 e 1518 bytes) e que sejam testados por um determinado período de tempo e um certo número de vezes. Isso porque todos estes tamanhos de quadros são utilizados na rede e, assim, todos os resultados precisam ser conhecidos.

Os testes mencionados na RFC 2544 são *Throughput* (Vazão), *Latency* (Latência), *Frame Loss* (Perda de Quadros) e *Back-to-back* (fim-a-fim). Neste trabalho serão utilizados apenas os testes de vazão e latência, que serão abordados nas próximas seções.

4.1.1 Vazão

A vazão de dados expressa a quantidade máxima de dados que podem ser transmitidos da origem ao seu destino. No entanto, a definição e medição da taxa de transferência são complicadas pela necessidade de definir um nível de qualidade aceitável. Por exemplo, se 10%

de quadros perdidos ou com erros forem considerados aceitáveis, então a vazão é medida a uma taxa de erros de 10%. Mas a definição geral é de que a vazão deva ser calculada sem erros ou quadros perdidos (BURGESS, 2004).

Burgess (2004, p.4, tradução nossa) explica a relação entre o valor nominal e o valor efetivo da taxa de dados:

> Em qualquer sistema Ethernet a capacidade máxima absoluta será igual à taxa de dados, por exemplo, 10Mbit/s, 100Mbit/s ou 1000Mbit/s. Na prática, estes números não podem ser alcançados devido às configurações dos quadros ethernet, que possuem campos adicionais para o transporte dos quadros. Os quadros de menor tamanho possuem uma taxa de transferência mais baixa do que os quadros de tamanho maior por causa da adição dos bytes do preâmbulo e do espaço entre os quadros, que não contam como dados.

A taxa de transferência máxima para os vários tamanhos de quadro segundo Burgess (2004, p.5), pode ser vista nos quadros abaixo:

Quadro 4.1 - Sistema 10Mbits/s

Tamanho do Quadro	Vazão de Dados	Preâmbulo e IGP	Quadros por segundo
64 bytes	7,62 Mbit/s	2,38 Mbit/s	14,88
128 bytes	8,65 Mbit/s	1,35 Mbit/s	8,45
256 bytes	9,27 Mbit/s	0,72 Mbit/s	4,53
512 bytes	9,62 Mbit/s	0,38 Mbit/s	2,35
1024 bytes	9,81 Mbit/s	0,19 Mbit/s	1,20
1280 bytes	9,84 Mbit/s	0,15 Mbit/s	0,96
1518 bytes	9,87 Mbit/s	0,13 Mbit/s	0,81

Fonte: BURGESS (2004, p.5)

Quadro 4.2 - Sistema 100Mbits/s

Tamanho do Quadro	Vazão de Dados	Preâmbulo e IGP	Quadros por segundo
64 bytes	76,19 Mbit/s	23,81 Mbit/s	148,81
128 bytes	86,49 Mbit/s	13,51 Mbit/s	84,46
256 bytes	92,75 Mbit/s	7,25 Mbit/s	45,29
512 bytes	96,24 Mbit/s	3,76 Mbit/s	23,50
1024 bytes	98,08 Mbit/s	1,92 Mbit/s	11,97
1280 bytes	98,46 Mbit/s	1,54 Mbit/s	9,62
1518 bytes	98,70 Mbit/s	1,30 Mbit/s	8,13

Fonte: BURGESS (2004, p.5)

Quadro 4.3 - Sistema 1000Mbits/s

Tamanho do Quadro	Vazão de Dados	Preâmbulo e IGP	Quadros por segundo
64 bytes	761,90 Mbit/s	238,10 Mbit/s	1488,10
128 bytes	864,86 Mbit/s	135,14 Mbit/s	844,59
256 bytes	927,54 Mbit/s	72,46 Mbit/s	452,90
512 bytes	962,41 Mbit/s	37,59 Mbit/s	234,96
1024 bytes	980,84 Mbit/s	19,16 Mbit/s	119,73
1280 bytes	984,62 Mbit/s	15,38 Mbit/s	96,15
1518 bytes	987,00 Mbit/s	13,00 Mbit/s	81,27

Fonte: BURGESS (2004, p.5)

O teste de vazão, segundo a RFC 2544, se inicia enviando um número específico de quadros a uma taxa de 100%. Se algum quadro for perdido nessa operação, o teste é repetido a uma taxa mais baixa, o teste deve ser executado até que a vazão máxima seja encontrada.

Segundo a RFC 2544 (1999, p.15, tradução nossa):

> Os resultados dos testes de vazão devem ser apresentados na forma de gráfico, sendo a coordenada X o tamanho do quadro, e a coordenada Y deve ser a taxa de vazão dos quadros. Devem existir ao menos duas linhas no gráfico. A primeira linha informando a taxa de quadros teórica para os vários tamanhos de quadro e a segunda deve ser o resultado dos testes.

4.1.2 Latência

Latência, também conhecido como *delay* (atraso) é o tempo total necessário para um quadro viajar da sua origem ao seu destino. O tempo total é a soma do atraso de processamento dos elementos da rede e o atraso de propagação ao longo do meio de transmissão (BURGESS, 2004).

A RFC 2544 indica que o teste de latência seja executado por 120 segundos para cada tamanho de quadro e repetido por 20 vezes. Os quadros devem ser enviados na vazão máxima suportada.

"O modelo de relatório de latência deve ser informado no formato de uma tabela com uma linha para cada um dos tamanhos de quadro testados. Deve haver colunas para o tamanho do quadro, e a taxa de latência executada" (RFC 2544, 1999, p.16, tradução nossa).

5 SIMULAÇÃO

Esta etapa do trabalho consiste em realizar os testes de vazão e latência envolvendo os protocolos de transporte TCP e SCTP. Para a realização de tais testes, optou-se pela utilização da simulação ao invés da montagem de um cenário real. O uso da simulação apresenta algumas vantagens em relação a montagem de um cenário real, tais como: menor custo, maior flexibilidade, facilidade no controle de detalhes, além de possibilitar a realização de experimentos de hardware e/ou software que não estejam disponíveis no mercado.

Como ferramenta para execução dos testes, a primeira seria a utilização do software OPNET, mas depois de testar a ferramenta, verificou-se que a mesma não possuía suporte ao protocolo SCTP. Desta forma, optou-se pela utilização do software *Network Simulator* (NS2), pois o mesmo possui agentes para a utilização do protocolo SCTP. O NS2 é simulador muito popular nos meios acadêmicos por ter seu código fonte aberto, ser gratuito e possuir grande material de apoio e pesquisa na internet.

5.1 Network Simulator (NS-2)

O NS (*network simulator*) é um simulador *open source* (código aberto) baseado em eventos discretos com foco na simulação de redes. O NS começou como uma variante do REAL *network simulator* em 1989 e evoluiu substancialmente com o passar dos anos. Em 1995 o NS foi desenvolvido através do projeto VINT (*Virtual InterNetwork Testbed*) que contou com o apoio da DARPA (*Defense Advanced Research Projects Agency*).

Segundo Gonçalves (2005, p.5) "o simulador NS2 oferece suporte à simulação de várias tecnologias de rede (com e sem fio), cenários baseados nos protocolos TCP, UDP e SCTP, diversas políticas de fila e caracterização de tráfego com diversas distribuições estatísticas."

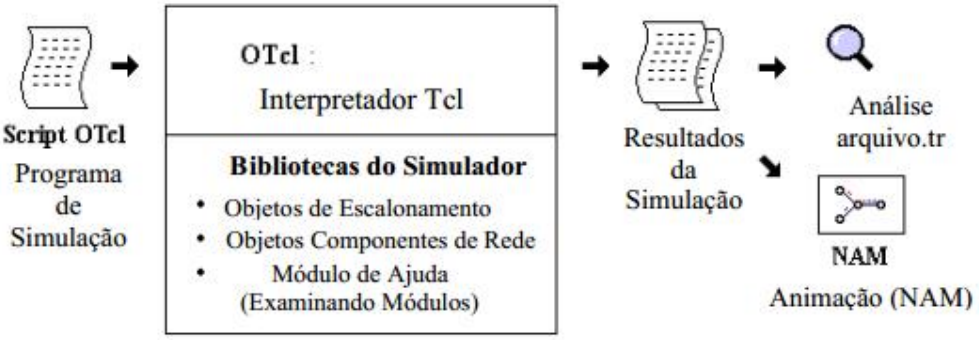
A programação do NS é feita em duas linguagens, C++ e OTcL (*Object Tool Command Language*), a primeira é utilizada para programar os módulos responsáveis pela execução dos protocolos e aplicações. A segunda é utilizada para definir as parametrizações dos módulos programados em C++. Desta forma, é possível, parametrizar, através de um *script* OTcL, um conjunto de módulos contidos nos diversos nós da rede num curto período de tempo. Além

disso, a alteração de um parâmetro no *script* OTcL não implica a compilação do código em C++ (OLIVEIRA, 2011).

A linguagem C++ é necessária para manipulação eficiente de *bytes, links,* cabeçalhos de pacotes, etc. Para redução do número de pacotes e do tempo de processamento dos eventos, o escalonador de eventos e os demais componentes de rede são escritos e compilados utilizando C++. Estes por sua vez, são disponibilizados para o interpretador OTcL, que cria uma correspondência para cada objeto em C++ (CHUNG e CLAYPOOL, 2007).

Para configurar e rodar uma simulação, o usuário deve escrever um *script* em OTcL que será interpretado pelo NS através de suas bibliotecas, obedecendo uma hierarquia de classes, cujo resultado final é a criação de dois arquivos que representam a análise dos resultados obtidos da simulação (INFORMATION SCIENCE INSTITUTE, 2001). A figura 5.1 ilustra este processo.

Figura 5.1 - Processo de criação e resultados da simulação



Fonte: Desenvolvido pelo autor, adaptado de Information Science Institute (2001).

6 EXPERIMENTAÇÃO

Após o exame dos protocolos TCP e SCTP, assim como a visão da importância do *benchmarking* e da simulação, este capítulo dá início a experimentação, onde será apresentado o ambiente utilizado, a metodologia empregada e todo o processo de simulação com o objetivo de realizar o estudo comparativo entre os protocolos TCP e SCTP.

6.1 Ambiente

Conforme descrito anteriormente optou-se pela utilização do NS-2, versão 2.35, que foi instalado em um microcomputador com as seguintes configurações de hardware e software:

- Sistema Operacional Windows 7 Professional 64 bits
- Processador Core i5 3330 3.0GHz
- 4Gb Memória
- HD Sata 500Gb
- Oracle VM VirtualBox 4.3.6 para execução da distribuição Ubuntu 13.10 (Linux)
- Simulador NS-2 versão allinone 2.35
- NAM (Network Animator) versão 1.15
- Microsoft Excel 2013

O sistema operacional instalado no microcomputador é o Windows 7 Pro 64bits, mas como o NS-2 foi originalmente desenvolvido para a plataforma Linux, optou-se pela utilização do Ubuntu 13.10. Se optou por utilizar o software Oracle VM VirtualBox 4.3.6, que emula sistemas operacionais dentro de máquinas virtuais, ou seja, com ele é possível rodar um sistema operacional dentro de outro, como o Ubuntu (ou qualquer outra distribuição de Linux) em um PC com Windows ou Mac, e vice-versa.

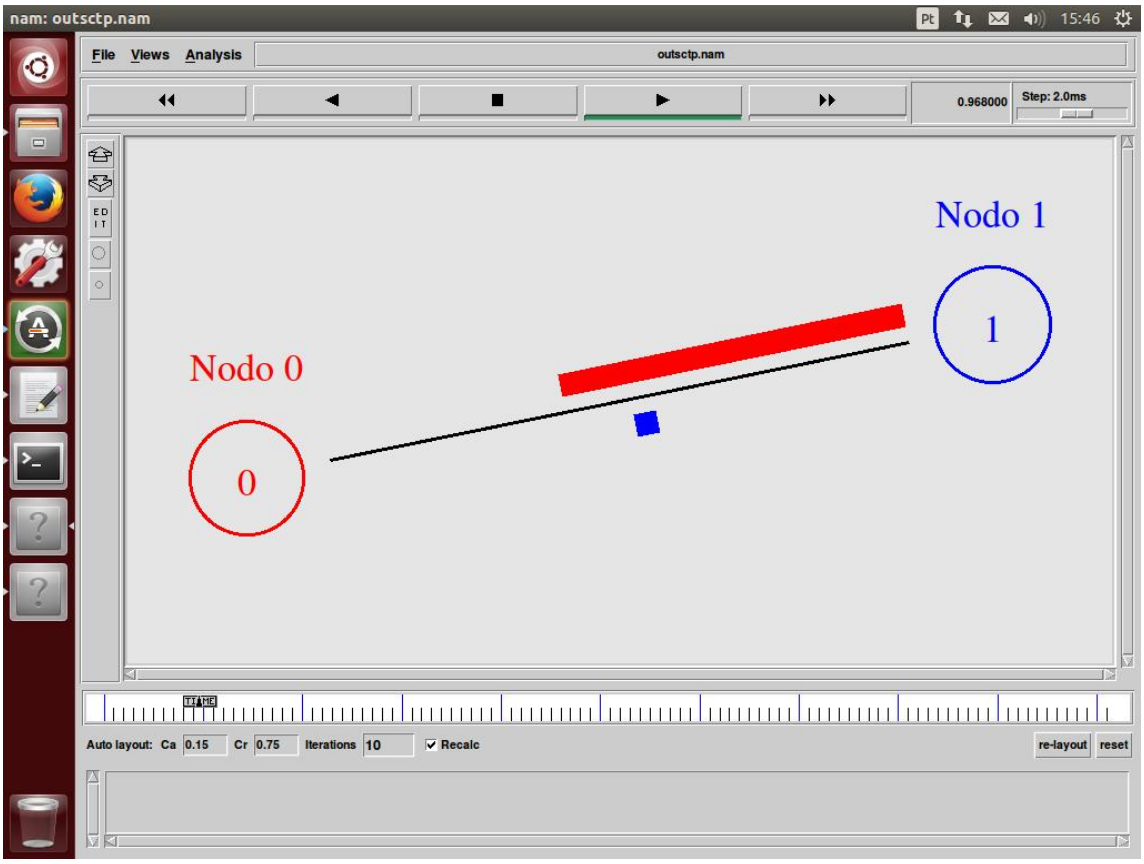
Além de instalar o NS-2, foi utilizado também a ferramenta NAM (Network Animator), que é um módulo opcional do NS-2. O NAM permite visualizar de maneira animada e ilustrativa a execução das simulações geradas. Já o Microsoft Excel 2013 foi utilizado para geração das tabelas e gráficos com base nos resultados obtidos nas simulações.

6.2 Metodologia

Após a instalação e configuração dos softwares necessários para a criação e execução das simulações, foi necessário realizar a escrita dos *scritps* na linguagem oTcL, tal arquivo precisa ser salvo com a extensão (.tcl), para posteriormente ser compilado pelo NS-2, via terminal Linux, através do comando ns nomedoarquivo.tcl.

Com o arquivo compilado pelo simulador, o *script* gera dois arquivos de saída, um deles com extensão (.nam) e outro com extensão (.tr). O arquivo (.nam) possibilita a utilização do *Network Animator* para a visualização dos resultados de maneira ilustrativa e animada, demonstrando virtualmente, como seria o funcionamento desta rede em ambiente real. A imagem 6.1 apresenta a interface do *Network Animator*.

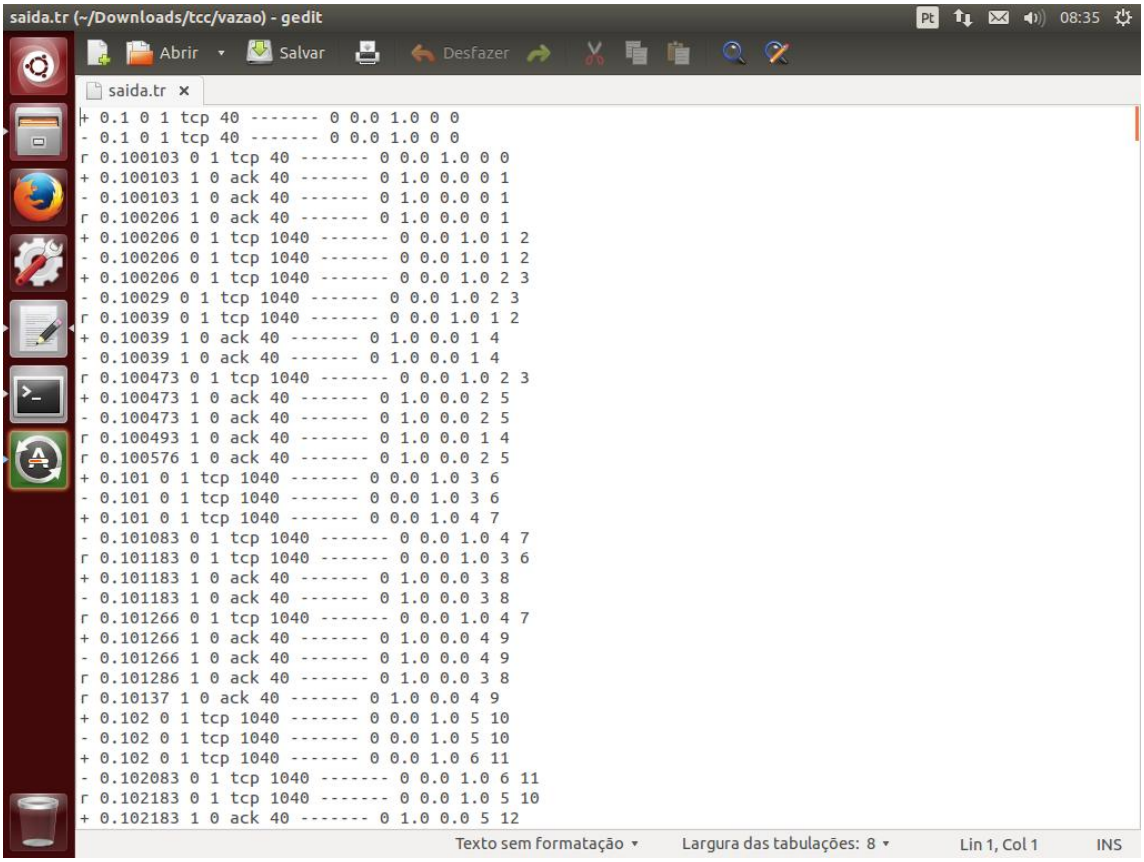
Figura 6.1 – Interface do Network Animator (NAM)



Fonte: Autor

O arquivo com extensão (.tr), mais conhecido como arquivo *trace*, apresenta detalhadamente em forma de texto todos os resultados da simulação. Cada evento gerado pelo *script* oTcl é salvo no arquivo *trace.* A figura 6.2 apresenta um exemplo do arquivo de saída (.tr).

Figura 6.2 - Arquivo com parte do resultado da simulação



Fonte: Autor

O arquivo *trace* possui 14 colunas, como pode ser visto na figura 6.2. Cada coluna do arquivo possui um significado, que pode ser observado na figura 6.3 abaixo:

Figura 6.3 - Significado de cada coluna do arquivo trace

OPERAÇÃO	TEMPO	NÓ PARTIDA PARCIAL	NÓ CHEGADA PARCIAL	PROTOCOLO	TAMANHO DO PAC	FLAGS	ID DO FLUXO	NÓ INICIAL	PORTA DE PARTIDA	NÓ DESTINO	PORTA DE CHEGADA	NÚMERO SEQUENCIAL	ID DO PACOTE
+	0.001	0	2	tcp	40	-------	1	0	.0	3	.0	0	0
-	0.001	0	2	tcp	40	-------	1	0	.0	3	.0	0	0
r	0.001132	0	2	tcp	40	-------	1	0	.0	3	.0	0	0
+	0.001132	2	3	tcp	40	-------	1	0	.0	3	.0	0	0
-	0.001132	2	3	tcp	40	-------	1	0	.0	3	.0	0	0
r	0.041452	2	3	tcp	40	-------	1	0	.0	3	.0	0	0

Fonte: Departamento de Engenharia Elétrica, Universidade Federal do Paraná (2011)

O quadro 6.1 apresenta uma descrição mais detalhada de cada coluna do arquivo *trace.*

Quadro 6.1 - Descrição do arquivo trace

Coluna	Nome	Descrição
1	Operação	r: pacote recebido
		d: pacote descartado
		e: erro
		+: pacote entrou na fila
		-: pacote saiu da fila
2	Tempo	Tempo que ocorreu o evento (em segundos)
3	Nó de Partida	ID do nó que enviou o pacote
4	Nó de Destino	ID do nó de destino do pacote
5	Protocolo	Protocolo de transmissão do pacote
6	Tamanho do Pac	Tamanho do pacote
7	Flags	C: ECN-echo
		P: pri_ (supposedly unused)
		A: Congestion Action
		E: Congestion Experienced (CE)
		F: Início rápido (Fast Start).
		N: ECN-capable
		SCTP-only: Somente SCTP.
8	ID do fluxo	Identificador do fluxo
9	Nó Inicial	Nó onde o pacote é gerado
10	Porta de Partida	Porta de partida do pacote
11	Nó Destino	Nó onde o pacote é recebido
12	Porta de Chegada	Porta de chegada do pacote
13	Número Sequencial	Número Sequencial
14	ID do pacote	Identificador único do pacote

Fonte: Desenvolvido pelo autor, adaptado de Information Science Institute (2001).

Com os dados do arquivo *trace* em mãos, foi necessário escrever outro *script*, desta vez na linguagem de programação AWK. Este *script* será utilizado para coletar apenas os dados necessários da pesquisa atual.

A palavra AWK é uma abreviatura das inicias dos sobrenomes dos criadores da linguagem (Alfred Aho, Peter J. Weinberger e Brian Kernighan). A linguagem AWK é interpretada linha por linha, muito utilizada na manipulações de arquivos de texto. A função básica do AWK é procurar linhas (ou outras unidades de texto) em arquivos que possuem certos

padrões especificados no programa. Para cada “padrão” deve haver uma ação associada, ou seja, quando uma linha corresponde a um dos padrões, AWK realiza a ação especificada naquela linha, em seguida, AWK continua processando as linhas de entrada desta maneira até encontrar o fim do arquivo de entrada.

Através do *script* criado na linguagem AWK foi possível coletar dados específicos do arquivo *trace,* como o tempo exato de envio e de chegada dos pacotes, quantidade total de pacotes enviados e recebidos, total de bytes recebidos.

O primeiro experimento realizado foi o teste de vazão sobre o protocolo TCP, realizado entre dois computadores, ligados diretamente, com uma conexão de 100Mbits/s e retardo de 0,1ms. O tipo de fila utilizado foi o *Droptail,* que representa o algoritmo FIFO (*First In, First Out*). Os testes foram realizados com quadros de 64, 128, 256, 512, 1024, 1280 e 1518 bytes, durante rajadas de 120 segundos cada, conforme exige a RFC 2544. Para se chegar a vazão máxima suportada, conforme exige a RFC 2544, foi utilizado um parâmetro que transmite um pacote a cada x tempo, ao final do teste, verificou-se se a quantidade de pacotes enviados e recebidos era o mesmo, ou seja, se não houve pacotes perdidos. No caso de perda de pacotes, o teste é reexecutado com um tempo x maior que o anterior, e assim sucessivamente até se encontrar a vazão máxima suportada.

Em seguida foram realizados os testes de vazão utilizando o protocolo SCTP, a metodologia e parâmetros utilizados para o teste de vazão do SCTP são os mesmos utilizados no teste de vazão do TCP, apenas substituindo o protocolo e os agentes utilizados no transmissor e no receptor dos pacotes.

Com os testes de vazão finalizados, iniciou-se o testes de latência com os protocolos TCP e SCTP. Para os testes de latência, se utilizou uma conexão de 100Mbits/s, retardo de 0,1ms e fila do tipo *Droptail.* Os testes foram executados na vazão máxima suportada para os quadros de 64, 128, 256, 512, 1024, 1280 e 1518 bytes, durante 120 segundos e repetidos por 20 vezes para cada tamanho de quadro, conforme exige a RFC 2544.

Os resultados encontrados nos testes de vazão e latência foram levados ao Microsoft Excel 2013, onde foram criados quadros e gráficos que pode ser visualizados no próximo capítulo.

6.3 Resultados

Após a realização das simulações e coleta dos dados obtidos foram geradas tabelas e gráficos para ilustrar os resultados. A seguir estão apresentados os resultados obtidos nos experimentos de vazão e latência dos protocolos de transporte TCP e SCTP.

6.3.1 Vazão

O quadro 6.2 apresenta os resultados dos testes de vazão utilizando o protocolo TCP. Na primeira coluna são apresentados os tamanhos dos quadros (em bytes) utilizados no testes, de acordo com a RFC 2544. A segunda coluna apresenta a taxa de transferência (em Mbits/s) obtida para cada tamanho de quadro. A terceira coluna apresenta a taxa de transferência máxima teórica (em Mbits/s), de acordo com BURGUESS (2004), conforme apresentado no capítulo 4.1.1, do referencial teórico.

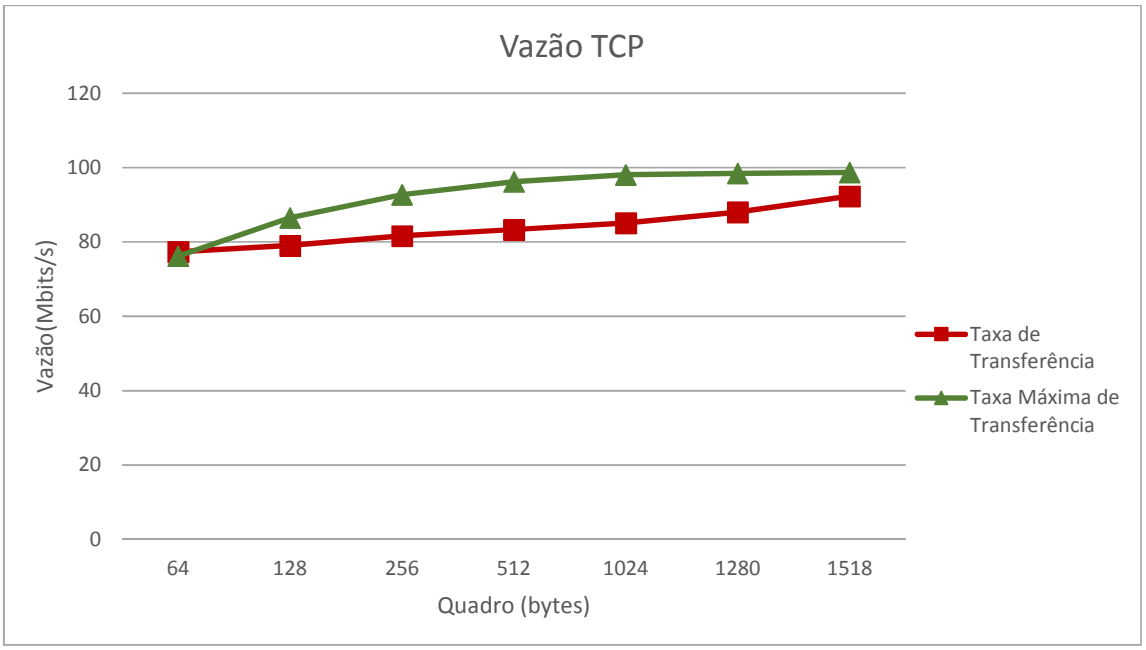
Quadro 6.2 - Teste de Vazão do Protocolo TCP

Tamanho do Quadro	Taxa de Transferência	Taxa máxima de Transferência
64	77,39	76,19
128	79,05	86,49
256	81,65	92,75
512	83,32	96,24
1024	85,12	98,08
1280	88,01	98,46
1518	92,32	98,7

Fonte: Autor

A figura 6.4 apresenta de maneira gráfica os resultados obtidos no teste de vazão do protocolo TCP.

Figura 6.4 - Gráfico de Vazão do Protocolo TCP



Fonte: Autor

O quadro 6.3 apresenta os resultados obtidos nos testes de vazão para o protocolo SCTP.

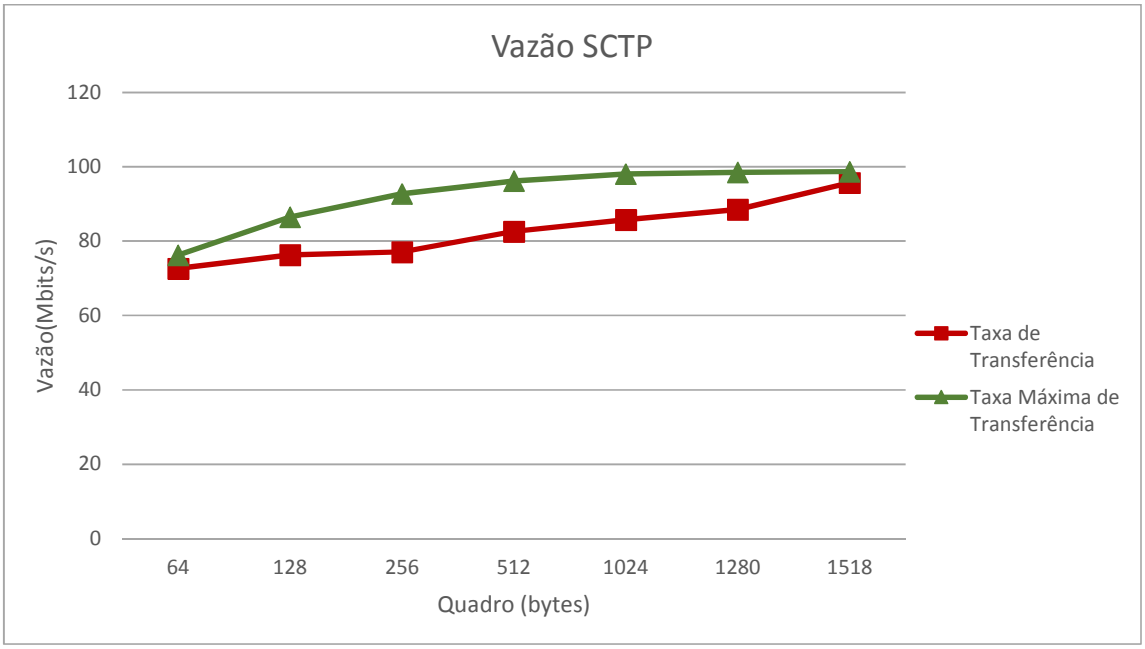
Quadro 6.3 - Teste de Vazão do Protocolo SCTP

Tamanho do Quadro	Taxa de Transferência	Taxa máxima de Transferência
64	72,7	76,19
128	76,27	86,49
256	77,12	92,75
512	82,63	96,24
1024	85,76	98,08
1280	88,53	98,46
1518	95,76	98,7

Fonte: Autor

Os resultados obtidos nos testes de vazão do protocolo SCTP podem ser visualizados de maneira gráfica na figura 6.5.

Figura 6.5 - Gráfico de Vazão Protocolo SCTP



Fonte: Autor

6.3.2 Latência

O quadro 6.4 apresenta os resultados dos testes de latência do protocolo TCP. A primeira coluna apresenta o tamanho dos quadros (em bytes) utilizados nos testes, de acordo com a RFC 2544. Já a segunda coluna apresenta o latência média (em milissegundos), que é o tempo médio necessário para o quadro viajar da sua origem ao seu destino.

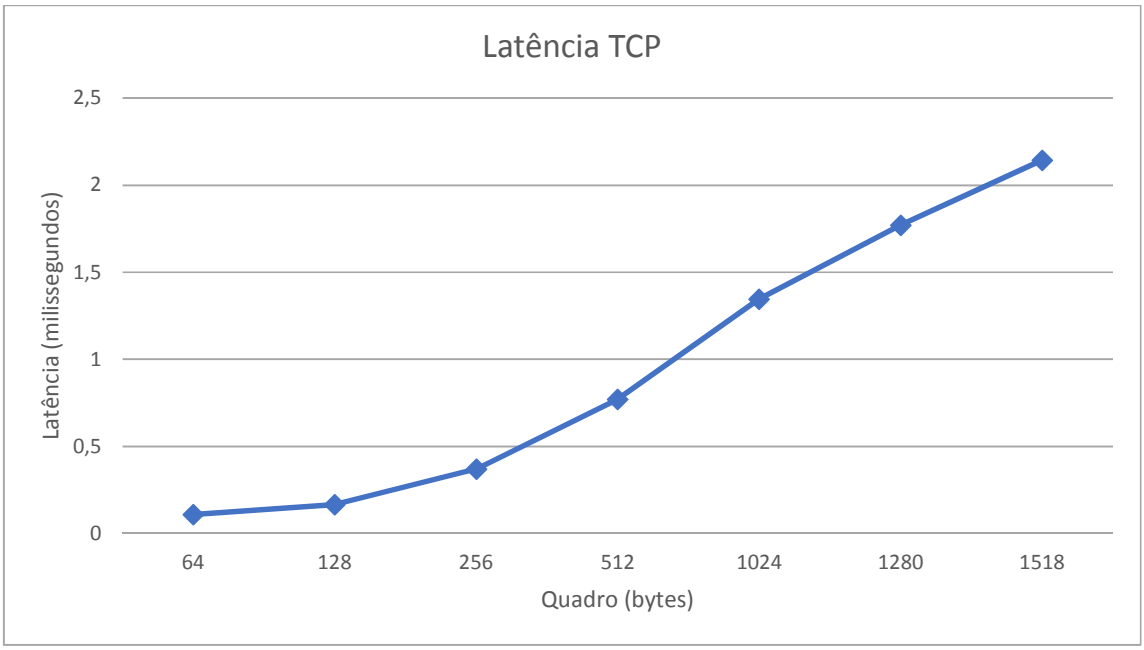
Quadro 6.4 - Teste de Latência do Protocolo TCP

Tamanho do Quadro	Latência Média
64	0,1083
128	0,1655
256	0,3703
512	0,7711
1024	1,3465
1280	1,7715
1518	2,1449

Fonte: Autor

A figura 6.6 apresenta por meio de um gráfico os resultados obtidos nos testes de latência do protocolo TCP.

Figura 6.6 - Gráfico de Latência Protocolo TCP



Fonte: Autor

O quadro 6.5 apresenta os resultados obtidos nos testes de latência do protocolo SCTP.

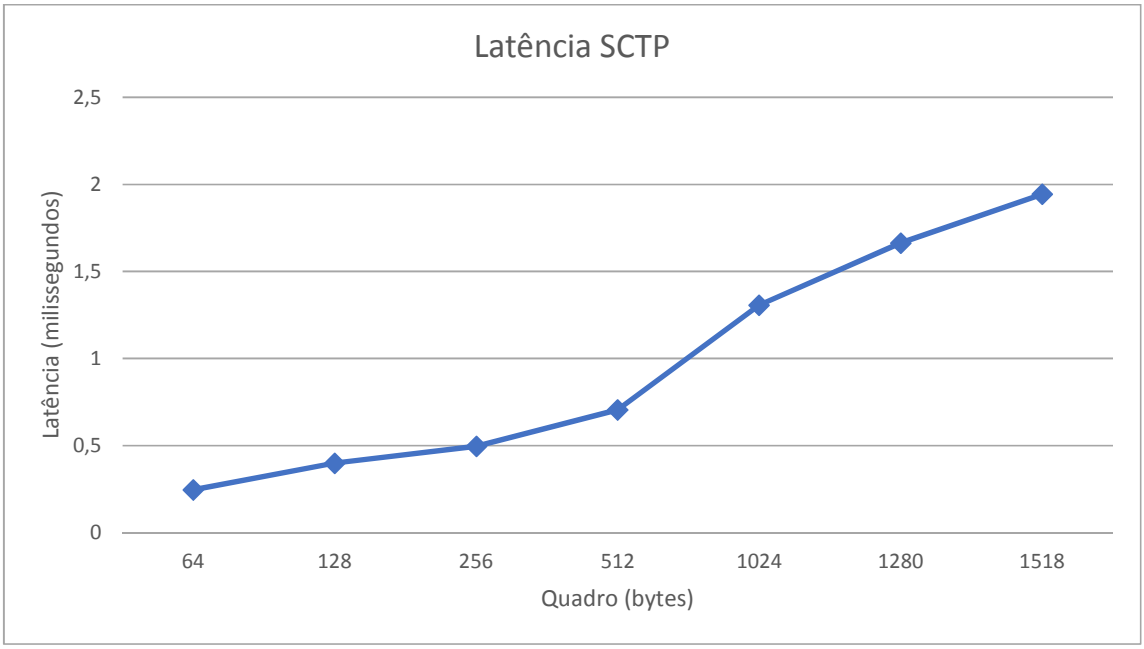
Quadro 6.5 - Teste de Latência do Protocolo SCTP

Tamanho do Quadro	Latência Média
64	0,246
128	0,3989
256	0,497
512	0,7055
1024	1,3065
1280	1,6652
1518	1,944

Fonte: Autor

A figura 6.7 apresenta de maneira gráfica os resultados dos testes de latência do protocolo SCTP.

Figura 6.7 - Gráfico de Latência Protocolo SCTP



Fonte:Autor

7 ANÁLISE DOS RESULTADOS

Com base nos resultados obtidos nas simulações, é possível fazer um estudo comparativo entre os protocolos TCP e SCTP. O quadro 7.1 apresenta os resultados obtidos nas simulações para os testes de vazão. A primeira coluna apresenta o tamanho do quadro (em bytes), a segunda e terceira coluna apresentam a vazão encontrada em Mbits/s, para os protocolos TCP e SCTP respectivamente.

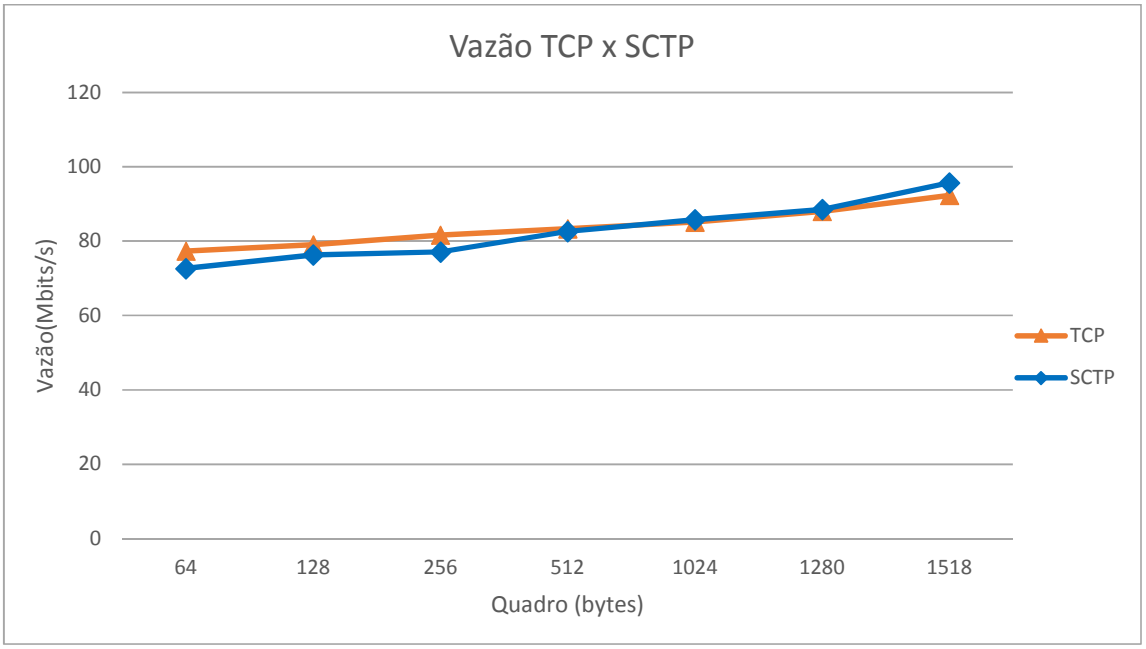
Quadro 7.1 - Quadro Comparativo de Vazão TCP e SCTP

Tamanho do Quadro	Vazão TCP	Vazão SCTP
64	77,39	72,7
128	79,05	76,27
256	81,65	77,12
512	83,32	82,63
1024	85,12	85,76
1280	88,01	88,53
1518	92,32	95,76

Fonte: Autor

A partir deste quadro, foi possível construir um gráfico, que apresenta melhor o desempenho de cada protocolo nos testes de vazão, este gráfico por ser visualizado na figura 7.1.

Figura 7.1 - Gráfico comparativo de Vazão entre TCP e SCTP



Fonte: Autor

A partir das informações apresentadas no quadro 7.1 e no gráfico comparativo, verificou-se que o protocolo TCP apresentou melhor desempenho quando utilizados tamanhos de quadros pequenos. A medida que o tamanho do quadro transmitido aumentava, percebeu-se que o desempenho do STCP frente ao TCP aumentava. Para os quadros de 64,128 e 256 o TCP obteve um desempenho em média 5% superior ao protocolo SCTP, já para os quadros de 512,1024 e 1280 os protocolos tiveram resultados bem semelhantes. Para o quadro de 1518 o SCTP obteve um desempenho 4% superior frente ao TCP.

Para encerrar a fase de análise de resultados foi elaborado o quadro 7.2, que apresenta os resultados dos testes de latência obtidos nas simulações. A primeira coluna apresenta o tamanho do quadro (em bytes), a segunda e terceira colunas apresentam a latência média (em milissegundos) encontrada para os protocolos TCP e SCTP respectivamente.

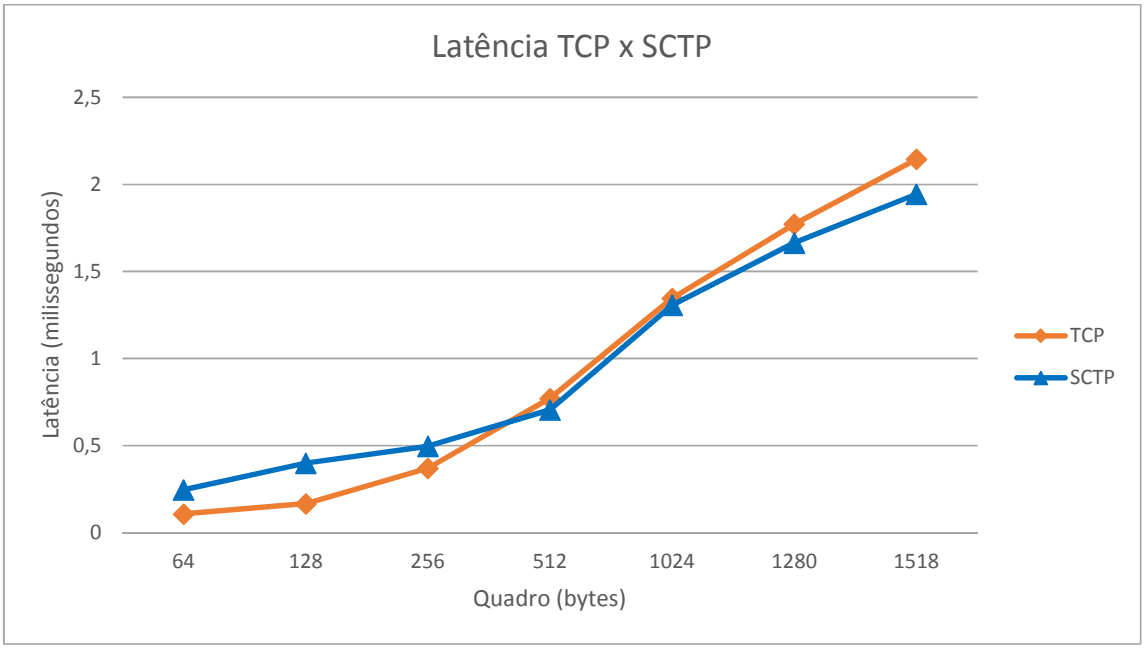
Quadro 7.2 - Quadro Comparativo de Latência TCP e SCTP

Tamanho do Quadro	Latência TCP	Latência SCTP
64	0,1083	0,246
128	0,1655	0,3989
256	0,3703	0,497
512	0,7711	0,7055
1024	1,3465	1,3065
1280	1,7715	1,6652
1518	2,1449	1,944

Fonte: Autor

A partir deste quadro, foi possível construir um gráfico, que apresenta melhor o desempenho de cada protocolo nos testes de latência, este gráfico por ser visualizado na figura 7.2.

Figura 7.2 - Gráfico Comparativo de Latência TCP e SCTP



Fonte: Autor

Assim como nos testes de vazão, ao criar a tabela e gerar o gráfico comparativo com os resultados dos testes de latência, foi possível observar que o protocolo TCP obteve melhor desempenho em quadros de tamanhos pequenos (64,128,256 bytes), enquanto que o protocolo SCTP obteve melhor desempenho nos quadros de tamanhos maiores (1280,1518 bytes). Para os quadros de 64 e 128 o TCP obteve um desempenho em média 57% superior ao SCTP. Para o quadro de 256 o TCP teve um desempenho 25% superior ao SCTP. Pode-se notar aqui, que a medida que o tamanho do quadro utilizado aumenta, o desempenho do TCP frente ao SCTP vai diminuindo. Para os quadros de 512 e 1024 o SCTP já possui melhor desempenho, obtendo em média um desempenho 5% superior ao TCP. Para os quadros de 1280 e 1512 o SCTP obteve um desempenho superior de 6% e 9% respectivamente.

Vale lembrar que para esse indicador, o que se deseja encontrar são valores baixos. Isto porque, ao contrário da vazão, onde os valores mais altos significam melhores resultados, para latência valores altos significam problemas, visto que esse indicador representa o tempo que a informação leva para chegar ao seu destino.

De acordo com os resultados apresentados, figuras 7.1 e 7.2, é interessante notar que nos testes de vazão e latência o protocolo TCP obteve um desempenho melhor para os quadros menores, enquanto que o SCTP obteve melhor desempenho para os quadros maiores, considerando que o maior quadro testado foi o de 1518 bytes. É importante citar aqui que todos os testes foram realizados com uma janela de recepção no valor 1, ou seja, para cada pacote enviado pelo transmissor, o receptor deve enviar uma mensagem de confirmação (ACK).

Também foram realizados experimentos com tamanhos de janela de recepção diferentes, mas não obteve-se resultados realistas. Os testes atuais foram realizados utilizando uma conexão de 100Mbits e um retardo de 0,1ms, quando alterado o tamanho da janela de recepção, o resultado permanecia o mesmo, independentemente do tamanho da janela definido. Descobriu-se que resultado não se alterava devido ao retardo ser muito baixo, foram então realizados testes com um retardo um pouco maior que os 0,1ms atuais. Nestes novos testes foram utilizados retardos entre 1ms e 2ms, porém, novamente não encontrou-se resultados realistas. Em alguns testes o resultado permanecia o mesmo, enquanto que em outros o desempenho de vazão e/ou latência diminuía em torno de 40% a 50%. Por este motivo e devido à falta de tempo para uma pesquisa mais aprofundada, decidiu-se não realizar os testes de vazão e latência com outros tamanhos de janela de recepção, que servirá de sugestão para trabalhos futuros.

CONCLUSÃO

O objetivo deste trabalho foi avaliar o desempenho do protocolo SCTP de forma comparativa com o protocolo TCP, que é atualmente o principal protocolo de transporte. Procurou-se fazer esse estudo de forma gradativa, dos aspectos históricos até a realização dos experimentos.

Para suportar a proposição deste trabalho, foi realizado um estudo sobre os modelos de referência OSI e TCP/IP, assim como suas camadas. Em seguida estudou-se os protocolos UDP e TCP, onde foi verificado que o UDP apesar de possuir uma comunicação rápida e flexível, não é um protocolo confiável. Já o TCP é um protocolo confiável, porém muitas aplicações consideram o TCP limitado e menos eficiente em certas ocasiões.

A necessidade de se encontrar um protocolo adequado para a transmissão de mensagens de sinalização telefônica, fez com que o protocolo SCTP fosse desenvolvido, protocolos como o TCP e UDP foram julgados inadequados para essa tarefa. Inicialmente o SCTP foi apenas utilizado para a troca de mensagens telefônicas na Internet, porém, as características e funções únicas do SCTP o tornaram atraente para inúmeras aplicações da Internet, o que resultou em sua incorporação ao grupo dos tradicionais protocolos de transporte da pilha TCP/IP.

Com o propósito de avaliar o desempenho do SCTP em relação ao TCP é que esse trabalho foi desenvolvido. Para realizar a avaliação comparativa entre os protocolos, foi estruturado um ambiente de testes no simulador NS-2, onde foram realizados testes de vazão e latência para cada protocolo.

Com os resultados obtidos nos testes foi possível observar a diferença de desempenho dos indicadores propostos, demonstrando que o TCP foi superior nos testes de vazão e latência na transferência de pacotes menores, enquanto o SCTP obteve melhor desempenho nos testes de vazão e latência na transferência de pacotes maiores. Existiu a preocupação, por parte do autor, em buscar referências claras quanto aos indicadores a serem utilizados nesta pesquisa, o que foi obtido através da utilização da RFC 2544 como referencial teórico para a definição dos indicadores para os testes de desempenho de rede e consequente apresentação dos resultados obtidos.

Quanto aos objetivos propostos para esse trabalho, conclui-se que foram atingidos. Pode-se concluir que o SCTP é um protocolo válido, que está demorando algum tempo para crescer além de seu confinamento nas interconexões das grandes companhias de telecomunicações.

Com mais de trinta anos de experiência no currículo, o TCP e o UDP não possuem algumas funções do SCTP, mas cada um forneceu peças e ideias para tornar o SCTP possível. Nem o TCP nem o UDP podem trabalhar em *multihoming* e ambos não possuem a capacidade de enviar informações para um endereço alternativo, se o principal não puder ser alcançado. Somente o SCTP pode trabalhar com *multistreaming,* que resolve o problema com redes que só podem gerenciar poucas conexões simultâneas. O TCP, concorrente mais próximo do SCTP, terá de ser melhorado ou pode se tornar obsoleto.

A ferramenta utilizada nas simulações deste trabalho (NS-2) mesmo sendo muito conhecida no meio acadêmico, apresenta algumas dificuldades para usuários iniciantes, especialmente na sua instalação e configuração. Também houve alguma dificuldade para se realizar a análise dos resultados, sendo necessário a escrita de um código em outro linguagem para interpretação dos resultados. Se sugere para desenvolvimentos de trabalhos futuros nessa área, que o pesquisador antes de realizar qualquer simulação via NS-2, leve em consideração o tempo de aprendizado da ferramenta e também estude ferramentas de apoio.

Apesar SCTP não ser muito conhecido, foi possível verificar que este protocolo possui muito potencial, por este motivo, como sugestões para trabalhos futuros, sugere-se:

- Realizar o estudo comparativo do SCTP com o TCP utilizando diferentes topologias de redes.
- Estudo comparativo do TCP com SCTP utilizando *multistreaming*.
- Realizar o estudo comparativo do SCTP com o TCP utilizando diferentes tamanhos de janela de recepção.
- Utilizar outras métricas de desempenho, como o *frame loss.*
- Realizar simulações entre os protocolos SCTP e UDP.

REFERÊNCIAS BIBLIOGRÁFICAS

AMER, P. et al.; **Load Sharing for the Stream Control Transmission Protocol.** Março, 2013. Disponível em: <http://tools.ietf.org/html/draft-tuexen-tsvwg-sctp-multipath-06> Acesso em: 15 out. 2013.

BRADNER, S.; MCQUAID, J.; **RFC 2544 – Benchmarking Methodology for Network Interconnect Devices**. IETF Network Working Group, 1999. Disponível em: <http://www.ietf.org/rfc/rfc2544.txt>. Acesso em: 30 out. 2013.

BURGESS, Nigel; **RFC 2544 Testing of Ethernet Services in Telecom Networks.** Novembro, 2004. Disponível em: <http://cp.literature.agilent.com/litweb/pdf/5989-1927EN.pdf> Acesso em 30 out. 2013.

CECHIN, Sérgio Luis; **Protocolo SCTP: Stream Control Transmission Protocol.** Dezembro, 2008. Disponível em: <http://www.inf.ufrgs.br/~cechin/Net/sctp/sctp.html> Acesso em: 11 out. 2013.

CHUNG, Jae; CLAYPOOL, Mark; **Ns by example.** 2007**.** WPI – Worcester Polytechnic Institute. Disponível em: <http://nile.wpi.edu/NS/> Acesso em: 06 nov. 2013.

COSTA, Daniel Gouveia. SCTP: Uma Alternativa aos tradicionais protocolos de transporte da Internet. Rio de Janeiro, 2005

COSTA, Daniel Gouveia. **Uma Arquitetura Baseada em SCTP e SIP para Suporte a Aplicações VoIP Móveis e a Especificação Formal do seu Módulo de Controle**. Maio, 2006. Disponível em: < ftp://ftp.ufrn.br/pub/biblioteca/ext/bdtd/DanielGC.pdf> Acesso em: 10 out. 2013.

DANTAS, M.; Tecnologia de rede de comunicação e computadores. Axcel Books, 2002.

DEPARTAMENTO DE ENGENHARIA ELÉTRICA, UNIVERSIDADE FEDERAL DO PARANÁ, **NS-2 Simulador de Rede**, 2011. Disponível em: < http://laplace.eletrica.ufpr.br/ns2.html> Acesso em: 10 mar. 2014.

FOROUZAN, Behrouz A.; Comunicação de Dados e Redes de Computadores. 4° edição, 2008.

FOROUZAN, Behrouz A.; FEGAN, Sophia Chung; **Protocolo TCP/IP**. 3° edição, 2009.

GONÇALVES, Lucas Coelho; CORRÊA, Marcos Estevo de Oliveira. **Tutorial de NS-2.**Julho, 2005. Disponível em: < http://www.midiacom.uff.br/~debora/redes1/pdf/tutorial-ns2.pdf > Acesso em: 04 nov. 2013.

INFORMATION SCIENCES INSTITUTE (ISI). **The Network Simulator - ns-2**. 2001. Disponível em: <http://www.isi.edu/nsnam/ns/>. Acesso em: 06 nov. 2013.

JONES, J. Tim; **Better networking with SCTP.** Fevereiro, 2006. Disponível em: <http://www.ibm.com/developerworks/library/l-sctp/> Acesso em: 15 out. 2013.

OLIVEIRA, Rodolfo; **Introdução ao simulador de redes The Network Simulator ns-2;** Faculdade de Ciências e Tecnologia - Universidade Nova Lisboa, entre 2010 e 2011. Disponível em: <http://tele1.dee.fct.unl.pt/mst_2010_2011/enuncs/ns_apoio.pdf> Acesso em: 05 nov. 2013.

PFÜTZENREUTER, Elvis; **Aplicabilidade e desempenho do protocolo de transporte SCTP;** Dezembro, 2004. Disponível em: <http://epx.com.br/mestrado/msc-epx-2004.pdf> Acesso em: 11 out. 2013.

PINHEIRO, José Mauricio Santos; TCP e Protocolos de Janelas Deslizantes; Abril, 2010. Disponível em: <http://www.projetoderedes.com.br/artigos/artigo_tcp_e_protocolos_de_janelas_deslizantes.php> Acesso em: 08 nov. 2013.

POSTEL, Jon. **Internet Protocol**, IETF RFC 791, Setembro, 1981. Disponível em: <http://www.ietf.org/rfc/rfc791.txt>. Acesso em: 15 set. 2013.

POSTEL, Jon. **User Datagram Protocol**, IETF RFC 768, Agosto, 1980. Disponível em: <http://www.ietf.org/rfc/rfc768.txt>. Acesso em: 20 set. 2013.

POSTEL, Jon. **Transmission Control Protocol**, IETF RFC 793, Setembro, 1981. Disponível em: <http://www.ietf.org/rfc/rfc793.txt>. Acesso em: 24 set. 2013.

PRODANOV, Cleber. **Manual de Metodologia Científica.** 3ª ed. Novo Hamburgo: FEEVALE, 2003. 79p.

STEWART, R. et al. **RFC 2960 – Stream Control Transmission Protocol**. IETF Network Working Group, 2000. Disponível em: <http://www.ietf.org/rfc/rfc2960.txt>. Acesso em: 06 ago. 2013.

STEWART, R. et al. **RFC 4460 – Stream Control Transmission Protocol (SCTP) Specification Errata and Issues**. IETF Network Working Group, 2006. Disponível em: <http://tools.ietf.org/html/rfc4460>. Acesso em: 07 out. 2013.

STEWART, Randall; METZ, Chris. **SCTP New Transport Protocol for TCP/IP.** Dec. 2001. Disponível em: <http://roland.grc.nasa.gov/nrg/local/sctp.net-computing.pdf>. Acesso: 21 ago. 2013.

TANENBAUM, Andrew S.; WETHERALL, David; **Redes de Computadores**. 5ª Edição, 2011.

TORRES, Gabriel; **Como o Protocolo TCP/IP Funciona.** Abril, 2007. Disponível em: <http://www.clubedohardware.com.br/artigos/Como-o-Protocolo-TCP-IP-Funciona-Parte-1/1351/1>. Acesso em: 27 set. 2013.

TORRES, Gabriel; **O Modelo de Referência OSI para Protocolos de Rede.** Abril, 2007. Disponível em: < http://www.clubedohardware.com.br/artigos/O-Modelo-de-Referencia-OSI-para-Protocolos-de-Rede/1349 >. Acesso em: 26 set. 2013.