ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO

ALEXANDRE JOSÉ BARBIERI DE SOUSA

***IXnet* – Proposta de Internet alternativa para
aplicações sensíveis a atraso.**

**São Paulo
2009**

ALEXANDRE JOSÉ BARBIERI DE SOUSA

***IXnet* – Proposta de Internet alternativa para
aplicações sensíveis a atraso.**

**Tese apresentada à Escola Politécnica da
Universidade de São Paulo, para obtenção do
título de Doutor em Engenharia Elétrica.**

Área de Concentração: Sistemas Eletrônicos

Orientador: Prof. Dr.Sérgio Takeo Kofuji

**São Paulo
2009**

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 31 de julho de 2009.

Assinatura do autor _____

Assinatura do orientador _____

FICHA CATALOGRÁFICA

Sousa, Alexandre José Barbieri de
IXNET – proposta de internet alternativa para aplicações sensíveis a atraso / A.J.B. de Sousa. -- ed.rev. -- São Paulo, 2009.
183 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Sistemas Eletrônicos.

1. Internet (Estrutura; Serviços; Qualidade) 2. Redes de computadores (Eficiência; Simulação) I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Sistemas Eletrônicos II. t.

AGRADECIMENTOS

Ao meu orientador, Professor Doutor Sérgio Takeo Kofuji, pela sabedoria e precisão nos detalhes.

Aos Professores Doutores Antonio Luiz Rigo e Wagner Luiz Zucchi pela organização das idéias e a melhor forma de expressá-las.

Aos colegas Leonardo Marchetti e Fernando Teubl pela colaboração.

À minha esposa Gisele, meu grande amor, pelo incentivo e excelentes sugestões.

Àqueles a quem dedico muito amor - meus pais Jair e Eneida, e minha avó Lenora, que não mediram esforços no apoio à minha formação pessoal e cultural.

RESUMO

A Internet pode não atender às necessidades exigidas por aplicações sensíveis a atraso tais como voz, vídeo ou até jogos em redes. A distância de saltos entre o cliente e o servidor, o congestionamento dos links e a aleatoriedade do caminho do pacote, podem causar atrasos. Existem motivações e até implementações de infra-estruturas alternativas para atender necessidades não garantidas pela Internet. Um exemplo é o projeto Rede COMEP, que propõe uma rede alternativa a fim de diminuir a distância e aumentar a capacidade entre os usuários (alunos e professores) e o conteúdo. Esta tese propõe uma infra-estrutura de rede de alta velocidade, exclusiva para aplicações sensíveis a atraso. A rede, denominada de IXnet, origina-se de experimentos de roteamento compartilhado por meio do IXP, estudo do tráfego de uma aplicação sensível a atraso e uso de simulador de rede. O uso da IXnet pode ampliar a frente de negócios para as operadoras e usuários que fazem uso de aplicações sensíveis a atraso, melhorando a qualidade na prestação de seus serviços.

Palavras-chave: ponto de troca de tráfego, jogos, atraso, servidor de roteamento; BGP; roteamento.

ABSTRACT

The Internet may does not meet all the demands required by applications sensitive to delay. Voice, video or even game applications demand low delay. The distance of hops between client and server, link traffic and the ramdomness of the package path, may cause delays. There is motivation and even implementation of alternative infra-structure to improve access. One example is the COMEP Network project, that proposes an alternative network in order to comply with the specific demands of such applications. This thesis proposes a high speed infra-structure network, exclusive for delay sensitive applications. This network, named IXnet, has it's origins in experiments with shared routing IXP, traffic studies of a sensitive to delay application and use of network simulator. The use of IXnet may increase the business front for operators and users that utilize sensitive to delay applications, improving the quality of the services provided.

Key-words: Internet Exchange Point; games; delay; router server; BGP; routing.

LISTA DE ILUSTRAÇÕES

Figura 1 – Topologia ilustrativa usando Ponto de Troca de Tráfego	15
Figura 2 – Etapas do trabalho	20
Figura 3 – Crescimento de usuários na Internet -isc.org (2008)	21
Figura 4 – Crescimento do MMORPG - MMOGCHART (2008)	23
Figura 5 – Tela de um jogo com atraso - <i>Gunbound World Champion</i> (2008)	27
Figura 6 – Conexão da Rede COMEP (rede Comep, 2009)	29
Figura 7 – Tela do Site usado para divulgar a url do jogo	31
Figura 8 – Exemplo de coleta do tráfego do jogo	32
Figura 9 – Tela do <i>Stream</i> - Acesso aos servidores do <i>Counter Strike</i>	34
Figura 10 – Diagrama de funcionamento do Trafip (www.telcomanager.com)	36
Figura 11 – Conectividade da Internet- IPv4 Internet Topology (2008)	37
Figura 12 – Exemplo de utilização do IXP Metro- IXP Metro(2008)	39
Figura 13 – IXP(S) distribuídos nos continentes (Packet Clearing House, 2009)	39
Figura 14 – Gráficos ilustrando relação de distância por atraso (Feldman, 2007)	40
Figura 15 – Troca de rotas no servidor de roteamento, Halabi e McPherson (2000)	41
Figura 16 – Exemplo de troca de tráfego entre os participantes. Sousa (2004)	42
Figura 17 – Roteadores vizinhos do BGP	48
Figura 18 – Exemplos de IBGP e E-BGP	49
Figura 19 – Módulos do equipamento Trafip	50
Figura 20 – Tela do programa HLTV	51
Figura 21 – Tráfego coletado para caracterização de tráfego- bits por segundo	52
Figura 22 – Pacotes coletados para caracterização de tráfego - pacotes por segundo	52
Figura 23 – Diagrama de fases da Simulação	58
Figura 24 – Execução do programa analisador	61
Figura 25 – Diagrama dos programas implementados	62
Figura 26 – Exemplo ilustrativo de um acesso pela Internet	63
Figura 27 – Topologia 1, cenário 1	65
Figura 28 – Tamanho de pacotes Cliente Servidor / Servidor Cliente	65

Figura 29	Tamanho do pacote do Cliente para Servidor	66
Figura 30	Tamanho do pacote do Servidor para o cliente	66
Figura 31	Topologia experimento 2 – cenário 1	68
Figura 32	Tela do programa analisador	69
Figura 33	Atraso pela quantidade de pacotes Cliente-Servidor	70
Figura 34	Atraso pela quantidade de pacotes Servidor-Cliente	71
Figura 35	Atraso pela quantidade de pacotes e saltos Cliente Servidor	72
Figura 36	Atraso pela quantidade de pacotes e saltos Servidor-Cliente	73
Figura 37	Atraso pela quantidade de pacotes Cliente Servidor	74
Figura 38	Comparação Cenário 1- Internet com e sem Congestionamento Cliente-Servidor	75
Figura 39	Comparação Cenário 1- Internet com e sem Congestionamento Servidor-Cliente	76
Figura 40	Exemplo ilustrativo de um acesso pela Internet usando um IXP	77
Figura 41	Topologia experimento 2 – cenário 2	78
Figura 42	Atraso pela quantidade de pacotes e saltos Cliente Servidor IXP simulado	79
Figura 43	Exemplo ilustrativo de acessos do cenário 3	80
Figura 44	Topologia experimento cenário 3	81
Figura 45	Topologia experimento BGP	83
Figura 46	Tabela de roteamento do protocolo BGP	83
Figura 47	Comparação da Internet / Ixnet do tráfego cliente para servidor	84
Figura 48	Comparação da Internet / Ixnet do tráfego servidor para cliente	85
Figura 49	Comparação de atraso da Internet / Ixnet do tráfego por usuário	85
Figura 50	Comparação geral da Internet / Ixnet do tráfego cliente para servidor	89
Figura 51	Comparação geral da Internet / Ixnet do tráfego servidor para cliente	90
Figura 52	Ilustrativa - Tendência IXnet	93

LISTA DE TABELAS

Tabela 1 – Características de Aplicações (Multirede, 2007)	26
Tabela 2 – Pesos de rotas e protocolos de roteamento (DEETC, 2003)	45
Tabela 3 – Acessos nos servidores de jogos	53
Tabela 4 – Consolidação da caracterização servidor cliente	54
Tabela 5 – Consolidação da caracterização cliente servidor	56
Tabela 6 – Bandas alocadas na simulação	60
Tabela 7 – Variáveis do primeiro experimento	64
Tabela 8 – Variáveis do segundo experimento do primeiro cenário	67
Tabela 9 – Média de tempo de resposta por continente (internettrafficreport.com , 2009)	69
Tabela 10 – Variáveis do primeiro experimento do segundo cenário	77
Tabela 11 – Amostra para simulação Cenário 1	86
Tabela 12 – Amostra para simulação Cenário 3	87

LISTA DE ABREVIATURAS E SIGLAS

AS	Autonomous System
BGP	Border Gateway Protocol
CS	Counter Strike
CIDR	Classless Inter-Domain Routing
FPS	First Person Shooter
IP	Internet Protocol
ISP	Internet Service Provider
IXP	Internet Exchange Point
NAM	Network Animator
MMORPG	Massively Multiplayer Online Role Playing Games
MPLS	Multiprotocol Label Switching
NAP	Network Access Point
NS	Network Simulator
PTT	Ponto de Troca de Tráfego
QOS	Quality of Service
RS	Router Server
RTS	Real Time Strategy games
SR	Servidores de Roteamento
TCL	Tool Command Language
TCP	Transfer Control Protocol
UDP	User Data Protocol

SUMÁRIO

RESUMO	05
ABSTRACT	06
LISTA DE ILUSTRAÇÕES	07
LISTA DE TABELAS	09
LISTA DE ABREVIATURAS E SIGLAS	10
1 – INTRODUÇÃO	13
1.1 Justificativa	14
1.2 Objetivo	16
1.2.1 Objetivos Específicos	16
1.3 Metodologia	17
1.4 Escopo	20
1.5 Organização do Trabalho	20
2 – REVISÃO DE LITERATURA	21
2.1 Importância da Internet	21
2.2 Importância dos jogos na Internet - Estudo de caso	22
2.2.1 Características dos jogos	24
2.2.2 Problema de lentidão nos jogos, vantagens e desvantagens dos métodos utilizados	24
2.3 Proposta de infra-estruturas de redes	28
3 – DESCRIÇÃO DO AMBIENTE PARA EXPERIMENTOS E SIMULAÇÃO PROPOSTA	31
3.1 Servidor do jogo	31
3.2 Analisador de Protocolos	32
3.3 Phyton	32
3.4 Simulador de Rede (NS-2)	33
3.5 Jogo escolhido – <i>Counter Strike</i>	33
3.6 Analisador de Protocolo – Wireshark	34
3.7 Uso de equipamento de coleta e caracterização de tráfego	35
3.8 Estrutura de Roteamento	36
3.8.1 Ponto de Troca de Tráfego na Internet (IXP)	36
3.8.2 Servidor de Roteamento	40
3.8.2.1 Servidor de Roteamento Zebra	42
3.8.3 Troca de tráfego direta entre participantes	43
3.8.4 Ponto de troca de tráfego híbrido	43
3.8.5 Pontos de acesso de alta velocidade – Ilha para acesso	44
3.8.6 Escalabilidade do IXP	44
3.9 Roteamento e Protocolo BGP	45
3.9.1 Conceitos de roteamento	45
3.10 Sistema Autônomo – AS	46
3.11 Protocolo de roteamento BGP v4	47
4 – CARACTERIZAÇÃO E SIMULAÇÃO DO JOGO NA INTERNET E NA REDE PROPOSTA	50
4.1 Caracterização do Tráfego	50
4.2 Coleta de dados	51
4.3 Caracterização do tráfego na LAN House	53
4.4 Fases da Simulação	57
4.5 Experimentos com o Simulador NS-2	59

4.6 Cenário Internet sem IXP	61
4.7 Primeiro Cenário - Simulação da Internet	63
4.8 Segundo Cenário - Simulação da Internet com um IXP	76
4.9 Terceiro Cenário – Simulação da rede proposta – IXnet	79
4.10 Implementando Protocolo BGP	82
5 ANÁLISE DE RESULTADOS	84
6 CONCLUSÃO	91
6.1 Recomendação IXNET	92
6.2 Trabalhos Futuros	93
7 REFERÊNCIAS	94
APÊNDICE A – Programa Gerador	97
APÊNDICE B – Programa Cliente Counter Strike	103
APÊNDICE C – Programa servidor Counter Strike	105
APÊNDICE D – Programa Analisador	107
APÊNDICE E – Programa BGP	111
APÊNDICE F – Caracterização do tráfego	116
APÊNDICE G – Configuração do servidor de Jogo Counter Strike	126
APÊNDICE H – Script Inicial do Python	128
APÊNDICE I – Configuração Experimentos Ponto de Troca de Tráfego	134
APÊNDICE J – Funcionamento do protocolo BGP, principais comandos de implementação e análise	144
ANEXO A – IXP (S) nos continentes	165

1 INTRODUÇÃO

O crescimento e a popularização da Internet possibilitaram o uso desta infraestrutura para diferentes aplicações que se comunicam com diversos usuários em localidades distintas. Exemplos destas aplicações são voz, vídeo, jogos de redes entre outras, as quais são exigentes quanto a baixo atraso, ou seja, podem ser prejudicadas caso a Internet apresente lentidão.

Esta tese aborda jogos em redes como ferramenta de estudo para aplicações sensíveis a atraso. A popularização dos jogos em rede e a possibilidade de coletar usuários aleatórios motivaram a escolha dos jogos neste trabalho.

Os Jogos em redes enfrentam problemas na Internet, geralmente ocasionados pelo atraso do pacote entre o jogador e o servidor ou ainda pela quantidade de acesso simultaneamente nos servidores de jogos. Para solucionar estes problemas o provedor de jogos pode aumentar a quantidade de servidores, porém não consegue controlar o atraso entre o usuário e o servidor.

Para entendimento do tráfego de jogos escolheu-se o *Counter Strike*, jogo popular do tipo FPS muito sensível a *atraso*. No *Counter Strike* o aumento de atraso pode decidir o vencedor da partida.

Foram realizados acessos aleatórios no servidor de jogos pela Internet, coletas em Lan Houses e posteriormente o tráfego foi observado através do analisador Wireshark (Ethereal) e do equipamento para caracterizar o tráfego chamado Trafip. O perfil de tráfego do jogo foi tratado e utilizado para simulação de rede através do simulador de rede *Network Simulator – NS-2*.

A partir dos experimentos realizados com IXP - *Internet Exchange Point* e a caracterização de tráfego foram implementados três cenários no simulador NS-2. O primeiro cenário simulou o acesso pela Internet para uso de uma aplicação sensível a atraso enquanto o segundo fez uso de um IXP exclusivo para estas aplicações. Posteriormente, no último cenário, simulou-se uma Internet alternativa somente com estruturas de pontos de tráfego interconectadas entre si.

A Internet especializada proposta denominada neste trabalho como IXnet, devido a redução de saltos e diminuição do congestionamento dos enlaces, possibilitará um tempo de resposta menor para os usuários, proporcionando também outras funcionalidades, tais como gerenciamento, segurança centralizada e escalabilidade.

1.1 Justificativa

O provedor das aplicações sensíveis a atraso consegue controlar sua rede interna, dimensionando seus servidores e links contendo, não consegue garantir uma baixa taxa de atraso para o seu cliente final, pois o pacote pode percorrer diversos caminhos aleatórios definidos pelo protocolo BGP, visto posteriormente. Diante deste fato, este trabalho propõe uma Internet especializada para aplicações sensíveis com o objetivo de diminuição do atraso entre o cliente e o servidor. Esta melhoria de atraso ocorre através da redução do número de saltos e controles de congestionamento.

A matéria “ **Uma visão otimista do futuro da Internet**”- Estado de São Paulo, 15 de março de 2009, no Caderno de Economia, revela alertas de especialistas sobre o risco de congestionamento em escala mundial da rede. Nesta mesma matéria, Bob Metcalfe, criador da rede Ethernet, e Jim Cicconi, vice-presidente da AT&T, prevêm que, com o crescimento do tráfego de dados e imagens no ritmo atual, em dois anos, o mundo poderá enfrentar um verdadeiro colapso da Internet. A matéria trata também da insegurança na Internet causada por vírus e fraudes, alegando que é mais fácil criar outra rede mais segura antes que aconteça uma catástrofe.

A rede proposta neste trabalho, denominada por seus autores de IXnet, é composta por de uma estrutura de troca de tráfego entre as operadoras que facilitará controles de segurança e gerenciamento, identificando possíveis ataques, além de seu foco principal que trata atrasos. Sua infra-estrutura é paralela à Internet, facilitando o processo de migração que poderia ocorrer de forma transparente e sua centralização facilitaria o gerenciamento e controle dos links e eventos de segurança.

Na figura 1 é possível observar uma rede exclusiva para usuários de aplicações sensíveis a atraso, em que os provedores trocam as rotas por meio do ponto de troca de tráfego e possuem conexão redundante pela Internet. Esta topologia poderá ser incrementada incluindo *firewalls* e/ou equipamentos redundantes como, por exemplo, dois servidores de roteamento, links, roteadores e novas funcionalidades, como gerenciamento e segurança centralizada.

A nuvem representada ao lado do usuário é a extensão da própria rede da operadora. Existe a possibilidade também de usar ilhas de acesso. Estas ilhas terão conexão direta até a operadora mais próxima e poderão servir para eventos de jogos ou salas para pesquisas, vídeo conferências, ou seja, uso de aplicações sensíveis a atraso.

Figura 1 – Topologia ilustrativa usando Ponto de Troca de Tráfego

Fonte: Próprio autor

A exploração do protocolo BGP e o ajuste dos filtros, visto no APÊNDICE J, proporcionaram a melhoria do tráfego na estrutura, alterando-se as políticas de roteamento de forma que as redes dos clientes fossem recebidas dos participantes. Essa alteração gera ganho de velocidade no acesso para os usuários.

1.2 Objetivo

Esta tese tem como objetivo propor uma Internet especializada para aplicações sensíveis a atraso a fim de proporcionar conexão rápida entre o servidor e o usuário final.

A Internet especializada proposta, denominada aqui como IXNET, se mostrará como uma infra-estrutura paralela com conexão destinada somente às aplicações selecionadas, conectando pontos de troca de tráfego entre si e somente tráfegos exigentes a atraso deverão ser roteados para a infra-estrutura.

Esta tese também teve como objetivo detalhar as características de tráfego do Counter Strike através de um processo de caracterização que poderá ser adaptado para novos tipos de tráfegos.

Adicionalmente, propõe-se simular e analisar o ambiente através de programas que poderão receber dados de roteadores, switches, servidores, usuários, índice de congestionamento, bem como proporcionar adaptações para novas simulações de diversas aplicações.

1.2.1 Objetivos Específicos

Estudo do tráfego do jogo Counter Strike, aplicação sensível a atraso escolhida como referência neste trabalho;

Caracterização do tráfego do jogo feita com uso de analisador de rede e equipamento para coleta e análise;

Implementação de programas para geração e análise do ambiente proposto no simulador NS2, possibilitando novas alterações/ simulações;

Comparação da rede proposta com a Internet especialmente pela redução do atraso;

1.3 Metodologia

A metodologia utilizada neste trabalho foi baseada na pesquisa exploratória. Segundo Mattar (1996), “a pesquisa exploratória visa prover o pesquisador de mais conhecimento sobre o tema ou problema de pesquisa em perspectiva” ou, ainda, “tem a finalidade de formular problemas e hipóteses para estudos posteriores”, Martins (2002).

O método usado foi o experimental, ou seja, “aquele em que variáveis são manipuladas de maneira pré-estabelecida, e seus efeitos suficientemente controlados. Tentam descobrir relações casuais. São precedidos de planejamentos experimentais, ou delineamento de experimentos”, Martins (2002).

Segundo Chizzotti (2001) o Método Experimental é aquele que submete um fato à experimentação em condições de controle, e o observa com critérios de rigor, mensurando a constância das incidências e suas exceções, admitindo como científico somente os conhecimentos passíveis de compreensão em condições de controle, legitimados pela experimentação e comprovados pela mensuração.

Para estudar uma aplicação sensível a atraso escolheu-se como referência o jogo *Counter Strike*, explicado posteriormente no item 3.5. O processo de análise envolveu tanto o usuário (acesso), como o servidor. A seguir as características que motivaram a escolha da aplicação Counter Strike:

- Execução em redes locais e Internet;
- Possibilidade de alterar quantidade de usuários;
- Programas sensíveis a atraso;
- Servidores de jogos abertos para alteração;
- Maior sensibilidade a *atraso* por ser um jogo FPS, ou seja, o *atraso* pode influenciar no vencedor da partida;

Após a escolha da aplicação, foi necessário gerar o acesso por usuários distribuídos de modo aleatório, ou seja, localizados em diversas origens e trilhando caminhos distintos na Internet.

Durante o intervalo e tempo que o servidor esteve disponível, observou-se que sua escolha estava relacionada ao baixo atraso do enlace e qualidade que o jogo oferecia para o usuário. Para conhecer o tráfego dos servidores inicialmente selecionou-se um analisador de protocolo. O processo de escolha desta ferramenta está descrito no item 3.6.

Para interpretação dos dados coletados pelo analisador, inicialmente, foi desenvolvido um *script* em Phyton (Apêndice H). A descrição das funcionalidades desta aplicação está descrita no item 3.3.

Pode-se considerar a análise dos pacotes a partir do Phyton de certa forma complexa devido à necessidade de filtrar o processo de sessão de todos os jogadores. Com o intuito de caracterizar melhor o tráfego do Counter Strike usou-se também o equipamento denominado Trafip.

O tráfego foi coletado pela Internet e em Lan House destinada somente para jogadores de Counter Strike, porém é definido um limite de acesso tanto no servidor, quanto na quantidade de usuários aleatórios que entravam no sistema. Para simular o aumento dos pacotes foi usado o NS-2, apresentado no item 3.4. O objetivo da análise no simulador é alternar cenários com as variáveis a seguir:

- Número de usuários;
- Tamanho dos links;
- Índice de congestionamento;
- Números de saltos do usuário até o servidor;
- Números de roteadores, servidores e clientes;

Posteriormente a simulação comparou:

- Cenário do uso da Internet;
- Cenário do uso do ponto de troca de tráfego;
- Cenário de Internet alternativa somente com pontos de troca de tráfego;

A metodologia utilizada e escolhida poderá ser adaptada em outra aplicação sensível a atraso e os programas gerados neste trabalho podem ser alterados para testar outras funcionalidades.

A seguir, um breve resumo das etapas do trabalho:

- 1 – A partir da estação do cliente no jogo escolher servidores na Internet com alto ou baixo atraso, analisando as diferenças do ponto de vista do jogador;
- 2 – Instalar o jogo com funcionalidade padrão e disponibilizá-lo na Internet com um link de baixa velocidade (600 kbytes de upload e 4 Megabyte de download), capturar o tráfego com o Wireshark e medir o atraso através da comparação das sessões;
- 3 – Transferir o servidor de jogos para uma rede com uma banda maior (Rede interna USP);
- 4 – Aperfeiçoar o ambiente do jogo, proporcionando mais acessos e posteriormente capturar o tráfego;
- 5 – Instalar o jogo em uma rede local e capturar os pacotes. Esta fase foi feita em uma Lan House;
- 6 – Consultar jogadores, artigos e empresas de jogos para analisar a tolerância quanto ao atraso;
- 7 – Caracterizar o tráfego através do Phyton e Wireshark e Trafip (equipamento de coleta);
- 8 – Usar o simulador de rede NS-2 para comparar um acesso do jogador para o servidor a partir de uma estrutura de troca de tráfego e pela Internet, gerando programas para simular e a analisar as variações para cada cenário;
- 9 – Comparar os resultados;

A seguir, na figura 2, um breve resumo das etapas do trabalho.

Figura 2 – Etapas do trabalho

Fonte: Próprio autor

1.4 Escopo

Esta tese tratou o atraso do pacote entre o servidor e o usuário final, considerando que os links entre os IXP não são congestionados, pois são de alta capacidade e com um gerenciamento centralizado.

A simulação foi baseada no jogo Counter Strike, pode-se testar outras funcionalidades alterando os programas gerados.

As bandas, atrasos, taxas de transmissão podem ser alteradas conforme a necessidade.

As simulações foram feitas com no máximo 350 roteadores, pode-se aumentar este número em novas simulações.

1.5 Organização do Trabalho

Para atender aos objetivos da pesquisa, o trabalho está estruturado em seis capítulos:

No capítulo 1, Introdução, apresentando a justificativa, objetivo, metodologia e escopo; Revisão de Literatura no capítulo dois; Descrição do ambiente para experimentos e simulação no capítulo 3; Caracterização e simulação no capítulo 4; Análise dos resultados no capítulo 5 ; Conclusão e Trabalhos Futuros no capítulo 6.

2 REVISÃO DE LITERATURA

2.1 Importância da Internet

A grande popularidade e disseminação que a Internet vem atingindo nos últimos anos têm despertado o interesse de diversos segmentos ligados à pesquisa e desenvolvimento, especialmente no que diz respeito à adequação do protocolo IP a uma série de aplicações como vídeo sob demanda, conferências, multimídias, TV interativa, tele medicina, etc. (Breu, 2007). Na figura 3 é possível observar o crescimento de usuários na Internet entre janeiro de 1994 e janeiro de 2008.

Figura 3 – Crescimento de usuários na Internet -isc.org (2008)

Fonte: Isc.org (2008)

Observa-se que no último ano houve um aumento de quase 100.000.000 usuários. As aplicações na Internet estão incorporando cada vez mais recursos que necessitam de conexão de tempo real. A infra-estrutura da Internet deve comportar um crescimento de tempo real e banda relativa a estas aplicações.

Existem motivações para uso de novas tendências como *Cloud Computing* e *SAAS- Software As A Service*, que suportam o compartilhamento de aplicações locais sensíveis a atraso em Datacenter(s). Estes Datacenter(s) necessitam de uma infra-estrutura de rede capaz de atender à demanda.

2.2 Importância dos jogos na Internet - Estudo de caso

A Internet tornou-se muito popular e uma das maiores e mais importantes aplicações em termos econômicos e culturais são os jogos de computadores. Atualmente a Internet é muito utilizada como infra-estrutura para os jogos em redes. O mercado de jogos cresce exponencialmente; de acordo com Anido e Penha (2004), o mercado de jogos eletrônicos nos EUA gera mais volume de receitas do que as bilheterias dos filmes de Hollywood. Os jogos da Internet permitem retornos financeiros para as empresas, seja ele por cobrança pelo jogo ou até venda de publicidade para os sites dos jogos.

Conforme mencionado anteriormente a sensibilidade dos jogos de redes em relação a atraso, motivou a sua escolha como estudo de caso.

De acordo com Lapoint e Winslow (2001), o rápido crescimento da Internet e queda de preço dos computadores iniciaram um crescimento de jogos multiusuários. Atualmente o público da Internet está mais popular, pois usuários que jogavam vídeo games passaram a usar também jogos na Internet que possibilitam interação real.

Segundo Silva et al. (2005), com o avanço das tecnologias de comunicação e o sucesso da Internet, foi possível implementar jogos, nos quais milhares de jogadores interagem simultaneamente e continuamente em um mundo virtual persistente, e por esse motivo, estes jogos foram classificados como massivamente multiusuário.

Segundo Feng et al. (2002), a recente explosão dos jogos multiusuários está começando a se tornar muito importante para entender o comportamento da rede e o uso para provisionar e desenhar uma infra-estrutura de rede futura. Uma rede que permita não só o crescimento, mas também a possibilidade de usuários de diferentes operadoras interagirem sem serem atrapalhados por atrasos.

Um número crescente de jogos é oferecido ao mercado, e vários são os recursos incorporados a cada dia. Para comprovar esse fato, Gardenghi et al. (2004) confirmam que com o aumento das conexões da Internet, os desenvolvedores de jogos de computador começaram a criar outros MMORPG(s)- Massively Multiplayer Online Role Playing Games, adicionando cada vez mais recursos.

Com aumento de disponibilidade de banda e diminuição dos preços, os jogos *on line* assumiram uma grande porcentagem do tráfego total da Internet, sendo este fato descrito por Breu (2007). A figura 4 retrata o crescimento de usuários dos jogos MMORPG.

Figura 4 – Crescimento do MMORPG - MMOGCHART (2008)

Fonte: MMOGHART (2008)

Buchleit (2004) recomenda como solução jogar em uma rede local; no entanto essa rede, geralmente é limitada pela distância e conseqüentemente por pequena concentração de interesse comum de usuários. Existem eventos que reúnem em uma rede diversos usuários para jogarem e um exemplo é o *Campus Party*.

2.2.1 Características dos jogos

Zhou, et al (2008) afirmaram que existem três categorias de jogos, *FPS - First Person Shooter*, *Massively Multiplayer Online Role Playing Games (MMORPG)* e *Real Time Strategy games (RTS)*. O atraso na Internet é impactante para jogos FPS (*First Person Shooter*) uma vez que pode determinar quem vence ou perde o jogo.

Breu (2007) observou que a caracterização do tráfego de rede do jogo *Counter Strike*, tem aspectos técnicos como tamanho do pacote e taxa envio e recebimento de pacotes, assim como características dos jogadores, como exemplo a origem dos pacotes e tempos de sessões. Claypool et al (2003) descrevem que os clientes do *Counter Strike* e servidores usam rajadas de pacotes pequenos, ou seja, a rede precisa atender esta demanda de pacotes mantendo um baixo atraso.

2.2.2 Problema de lentidão nos jogos, vantagens e desvantagens dos métodos utilizados

O acesso dos jogadores pela Internet enfrenta dois problemas: o atraso e o grande número de acessos simultâneos pelo servidor. Existem aplicações na Internet que são exigentes quanto à vazão e atraso, como, por exemplo, a estrutura MMORPG.

Henderson e Bhatti (2003) avaliaram o nível de tolerância do jogador conforme o atraso a que ele fica exposto e comprovaram que usuários com atrasos maiores não permanecem no jogo.

O aumento do número de jogadores e de aplicativos exige que os jogos estejam funcionando em uma rede que não apresente atraso, proporcionando um tempo de resposta eficiente da rede e uma estrutura de servidores que suporte esta quantidade de acesso.

Segundo Haishu (2006), os provedores de serviço na Internet estão cada vez mais preocupados com o atraso. O provedor de Internet deve conhecer o tráfego dos jogos a fim de prover uma estrutura de rede. A preocupação dos provedores ISP (*Internet Service Provider*) não é só prover acesso à Internet, mas também tratar o tráfego do cliente, porém mesmo usando diversas técnicas, às vezes, o atraso não está na rede deste provedor e sim no caminho que o pacote percorre até chegar ou retornar à origem.

Para entender como o atraso pode afetar uma aplicação é importante entender a causa do atraso. O atraso pode ser atribuído a fatores como: compressão, descompressão, fatores de redes, servidores e aplicação no servidor.

De acordo com Färber (2002), a rede é a primeira causa do atraso, ou seja, tem grande chance de ser a maior causadora deste, maior que o atraso do hardware e software do servidor. O atraso pode ser uma somatória de atrasos fim-a-fim, ou seja, atrasos de enfileiramento, propagação, transmissão e processamentos gerados em uma conexão do início ao fim.

O atraso entre dois pontos em uma rede IP pode ser medido de duas maneiras, atraso de ida (*one way delay*) ou atraso de ida-e-volta (*round-trip delay*). Nas duas medições considera-se o caminho pelo qual o pacote percorre, assim como sua priorização nas filas e CPU(s) dos roteadores. Esta tese considerara o round trip delay.

A seguir, a Tabela 1 (Multirede, 2008) referencia a média do tamanho de pacote, média de atraso entre pacotes, banda mínima e tempo para envio de diferentes aplicações.

Tabela 1 – Características de Aplicações (Multirede, 2007)**Fonte:** Multirede (2007)

Aplicação	Tamanho do pacote	Média Atraso entre pacotes	Banda Mínima	Tempo para enviar 1 pacote (256 Kbps)
Dados	Variável, até 1500 bytes	Variável	Variável	47 ms
Videoconferência	700 bytes (médio)	30 ms	160 Kbps	22 ms
Voz sobre IP	60 bytes	20 ms	24 Kbps	2 ms

No jogo Counter Strike percebeu-se que a faixa está próxima da aplicação Voz sobre IP, ou seja, a largura de banda é baixa a moderada, a perda de pacotes é alta, o atraso e sua variação é alta.

Shaikh et al. (2004) relataram que os jogos sensíveis a atraso geralmente produzem uma alta quantidade de tráfego na Internet. Os provedores de Internet não se preocupam em prover uma infra-estrutura compartilhada de servidores para suportar jogos. Geralmente a estrutura física é única e compartilhada para outras aplicações. O atraso pode gerar desconexão do usuário no servidor de jogos, este atraso gera a desistência dos jogadores.

Na estrutura [MMORPG](#), usa-se o termo atraso para identificar o período de tempo em que a estação do usuário fica atrasada em relação ao servidor. Geralmente, este atraso é causado pela baixa taxa de transferência da conexão ou pelo excesso de usuários no servidor. Na figura 5 pode-se observar a mensagem do jogo que informa o parâmetro atraso para os jogadores.

Figura 5 – Tela de um jogo com atraso -*Gunbound World Champion* (2008)
Fonte: *Gunbound World Champion* (2008)

Segundo Breu (2007), os servidores de jogos também são programados para derrubar os jogadores com atraso acima de 100ms automaticamente. Este fato impede que usuários permaneçam em alguns jogos que exigem pouco atraso, ou seja, o sistema do jogo impede a presença de jogadores que não possuem condições mínimas de rede (pouco atraso), necessárias para o jogo.

Nos jogos que usam recursos de voz, a exigência quanto ao atraso é ainda maior; segundo Breu (2007), a maioria dos jogos usam funcionalidades de voz sobre IP. Desta forma, além de um baixo atraso, o jogador precisa ter uma rede que atenda a requisitos para qualidade de voz, por exemplo, nos quais a variação do atraso não pode ser grande.

Shaikh et al. (2004), sugeriram um protótipo de infra-estrutura de rede para jogos; contudo, não se aprofunda em aspectos de roteamento.

De acordo com Breu (2007), com o aumento do tráfego de rede e a importância do mercado de jogos, o tráfego coletado e analisado pode ser usado como modelo para desenvolver funcionalidades de QoS- *Quality of Service*. Desta forma, novas classes de qualidade de serviço podem ser implementadas a partir do conhecimento do tráfego de um jogo, proporcionando uma melhoria no atraso de rede. Porém, para se ter este efeito deve-se implementar a política desejada de

QOS no caminho percorrido pelo pacote, ou seja, a rede de todos os provedores que apresentem atraso.

Lapointe e Winslow (2001), relataram que técnicas incluindo *caching*, agregação, filtragem e comutação inteligente são usadas para resolver os problemas de atraso. Estas técnicas podem ser executadas através de *hardwares* específicos gerando otimizações/configurações e suporte especializado. Um exemplo destas técnicas, apresentado também por Lapointe e Winslow (2001), propõe inserir caixas chamadas *booster boxes* na rede; porém não é recomendado colocar mais de um equipamento, pois pode aumentar a taxa de falhas e também não impede o atraso no percurso do pacote até chegar ou retornar nesta rede.

A abordagem tradicional usada pela maioria dos provedores de jogos é instalar uma infra-estrutura de servidores dedicada a cada jogo, resultando em um alto custo. Desta forma, os provedores de jogos conseguem dimensionar seus servidores para um grande contingente de usuários. Porém dependem do dimensionamento dos links de acesso com um baixo atraso fim-a-fim.

Feng et al. (2002) observaram que os jogos *on-line* requerem pouco atraso e comunicação ponto a ponto, bem como canais diretos de *broadcast* para proporcionar o tempo real. Este seria o cenário ideal, pois o provedor de jogos poderia dimensionar sua estrutura de servidores conforme o acesso e contar com pouco atraso do usuário até seu servidor. Porém o artigo não trata qual seria o esforço para que as operadoras passassem a usar conexões ponto a ponto entre elas e pontos centrais de trocas de tráfego.

Este trabalho difere de Lapointe e Winslow (2001) que apresentaram técnicas de comutação e de Breu (2007) que propôs funcionalidades de QOS dentro da rede de um único provedor.

2.3 Proposta de infra-estruturas de redes

O projeto da rede COMEP surgiu para atender necessidades específicas não garantidas pela Internet.

A rede COMEP é uma infra-estrutura de redes metropolitanas de ensino e pesquisa de alta capacidade para comunicação que interconecta a RNP- Rede Nacional de Pesquisa ao IBICT- Instituto Brasileiro de informação em Ciência e Tecnologia e ao LNCC- Laboratório Nacional de Computação Científica. Sua intenção é a diminuição da distância, o aumento da capacidade entre os usuários (alunos, professores) e o conteúdo (bibliotecas, serviços de informação, previsão de clima, soluções em grades para pesquisa de nanotecnologia e genômica entre outras). Vê-se na Figura 6 a topologia da Rede Comep.

Figura 6 – Conexão da Rede COMEP (rede Comep, 2009)
Fonte: Rede Comep (2009)

Na topologia da rede COMEP pode-se observar os enlaces em laranja que são em Gigabit(s) enquanto as demais são em Megabit(s).

Um outro movimento é o REMAV- Projetos de Redes Metropolitanas de Alta Velocidade que tem como objetivo promover em diversas regiões do país, a criação de infra-estrutura e serviços de redes de alta velocidade.

A REMAV usa o conceito de consórcio, com sucesso semelhantes em todo o mundo. A existência de interesses e objetivos comuns, levam instituições proverem o investimento e gestão da infra-estrutura para trocar/ compartilhar informações.

A Internet 2 também vem como uma proposta de aumentar sua velocidade porém, por enquanto, está restrita a poucas pessoas: somente pesquisadores e funcionários das faculdades e empresas patrocinadoras podem usá-la. Além de pesquisas voltadas para a área acadêmica, a iniciativa inclui a transferência, para o setor comercial, de tecnologias desenvolvidas e testadas ao longo da execução dos projetos.

A redução do custo dos enlaces e aumento das bandas de comunicação motivam iniciativas para uso de conexões físicas dedicadas a troca de tráfego. Um exemplo disto é o uso do enlace de 10 Gigabit.

3 DESCRIÇÃO DO AMBIENTE PARA EXPERIMENTOS E SIMULAÇÃO PROPOSTA

A seguir, uma descrição do ambiente, ferramentas de análise e simulação bem como os conceitos tratados nesta tese.

3.1 Servidor do jogo

O servidor escolhido para hospedar o jogo tem as seguintes características: Processador AMD Dual Core 1.7 GHZ / 2 Gigabytes de memória RAM, HD 110 GB, Windows XP. O servidor estará dedicado para o jogo e não poderá sofrer interrupções durante o período de captura. Alguns fatores que podem eventualmente causar impacto de desempenho do servidor devem ser minimizados.

Foram implementadas ações de segurança no servidor, tais como ativação de *Firewalls*, atualização de *patches* do Sistema Operacional, antivírus, mapeamento de portas de acesso.

Na figura 7 pode-se observar a tela do site divulgando o jogo. A URL deste site foi cadastrada em comunidades de jogos na Internet.

Figura 7 – Tela do Site usado para divulgar a url do jogo

Fonte: Próprio autor

3.2 Analisador de Protocolos

O analisador escolhido foi o Wireshark (Ethereal) que será descrito no item 3.6. O Jogo utiliza UDP como protocolo de transporte, portanto captura-se somente pacotes UDP. Além disto, limitou-se os arquivos ao tamanho de 50 Megabytes. Na Figura 8 uma amostra do log do Wireshark (Ethereal) capturada durante o jogo.

No. -	Time	Source	Destination	Protocol	Info
1	0.000000	89.212.6.48	201.81.88.110	UDP	Source port: 1507 Destination port: 27015
2	0.714948	201.81.88.110	89.212.6.48	UDP	Source port: 27015 Destination port: 1507
3	0.436617	89.41.232.183	201.81.88.110	UDP	Source port: 1823 Destination port: 27015
4	1.138378	201.81.88.110	89.41.232.183	UDP	Source port: 27015 Destination port: 1823
5	2.873159	93.81.39.205	201.81.88.110	UDP	Source port: 1755 Destination port: 27015
6	3.597714	201.81.88.110	93.81.39.205	UDP	Source port: 27015 Destination port: 1755
7	2.924201	64.72.116.55	201.81.88.110	UDP	Source port: 50956 Destination port: 27015
8	2.927524	201.81.88.110	68.142.64.166	UDP	Source port: 26901 Destination port: 27014
9	2.927531	201.81.88.110	64.72.116.55	UDP	Source port: 27015 Destination port: 50956
10	3.080961	64.72.116.55	201.81.88.110	UDP	Source port: 50956 Destination port: 27015
11	3.785239	201.81.88.110	64.72.116.55	UDP	Source port: 27015 Destination port: 50956
12	3.240917	64.72.116.55	201.81.88.110	UDP	Source port: 50956 Destination port: 27015
13	3.989136	201.81.88.110	64.72.116.55	UDP	Source port: 27015 Destination port: 50956
14	3.444577	64.72.116.55	201.81.88.110	UDP	Source port: 50956 Destination port: 27015
15	4.161230	201.81.88.110	64.72.116.55	UDP	Source port: 27015 Destination port: 50956
16	4.803953	87.126.243.215	201.81.88.110	UDP	Source port: 34641 Destination port: 27015
17	5.524309	201.81.88.110	87.126.243.215	UDP	Source port: 27015 Destination port: 34641
18	8.810482	116.14.28.164	201.81.88.110	UDP	Source port: 37725 Destination port: 27015
19	9.518988	201.81.88.110	116.14.28.164	UDP	Source port: 27015 Destination port: 37725
20	9.521117	201.37.228.78	201.81.88.110	UDP	Source port: 2859 Destination port: 27015
21	10.239525	201.81.88.110	201.37.228.78	UDP	Source port: 27015 Destination port: 2859
22	12.636382	201.81.88.110	68.142.64.166	UDP	Source port: 26901 Destination port: 27014
23	13.405595	212.33.23.118	201.81.88.110	UDP	Source port: 1317 Destination port: 27015
24	14.124499	201.81.88.110	212.33.23.118	UDP	Source port: 27015 Destination port: 1317
25	14.670188	62.44.213.224	201.81.88.110	UDP	Source port: 61530 Destination port: 27015
26	15.377820	201.81.88.110	62.44.213.224	UDP	Source port: 27015 Destination port: 61530
27	16.394022	89.136.80.224	201.81.88.110	UDP	Source port: 4044 Destination port: 27015
28	17.132390	201.81.88.110	89.136.80.224	UDP	Source port: 27015 Destination port: 4044

Figura 8 – Exemplo de coleta do tráfego do jogo

Fonte: Próprio autor

3.3 Python

Python é uma linguagem interpretada, orientada a objetos e possui ainda uma sintaxe simples, próxima da linguagem matemática, do MATLAB. Em Python, diferentemente de C++ ou Java, as funções são tratadas como objetos, característica de linguagens de programação funcional como Lisp, muito utilizada em aplicações de inteligência artificial. O Python foi utilizado no trabalho para interpretar as informações do Analisador de Protocolos.

3.4 Simulador de Rede (NS-2)

O NS-2 é um programa que pode gerar tráfego de rede na simulação. Suporta vários tipos de tráfego, *real time* e vídeo. Tem seu código aberto, e os vários usuários podem colaborar com seu desenvolvimento. O simulador NS-2 pode trabalhar com protocolos, em redes LAN e WAN; pode ainda simular tanto o TCP quanto o UDP. Teve sua primeira versão em 1989 e atualmente está na segunda versão.

O NS-2 pode ser executado no *Linux* e no Windows através da aplicação emulador *Cygwin*. O ambiente de uma rede pode ser analisado através da criação de um cenário, permitindo informações como topologia, *hosts*, configurações específicas do link e até comportamentos específicos da aplicação utilizada.

3.5 Jogo escolhido – *Counter Strike*

O *Counter Strike* é uma variante do *Half-Life* muito popular nos jogos de computador. Trata-se de um personagem em primeira pessoa baseado em rodadas nas quais equipes de contra-terroristas e terroristas entram em combate até a vitória. É necessário estratégia, trabalho em equipe e habilidade para ser um vencedor. É acessado através da ferramenta *Stream*. Por volta de 2005/2006 foi disponibilizada a versão do *Counter Strike Source*, que possui melhorias gráficas, porém um pouco mais lenta que a versão anterior.

O *Counter-Strike* foi um dos responsáveis pela massificação dos jogos em rede no início do século, sendo considerado o grande responsável pela popularização das casas de jogos (*LAN Houses*) em vários países.

O Jogos baseados em *Half Life* (*Counter Strike*) tornam-se praticáveis com atrasos inferiores a 150 milissegundos. O atraso pode influenciar o vencedor da partida. Desta forma, os jogadores tendem a escolher servidores onde há atraso abaixo de 50ms (levando em conta a localização geográfica dos servidores), vale lembrar que o atraso pode estar relacionado ao tempo do pacote percorrer da origem ao destino.

Pode-se observar na figura 9 que os servidores são listados com o valor do atraso entre o jogador e o servidor, informado no campo denominado Latency.

The screenshot shows the 'Servers' window in the Stream game client. It displays a list of 1162 servers. The columns shown are Game, Players, Map, and Latency. The servers listed are primarily Counter-Strike servers with various maps like de_dust2, de_dust2_fundo, de_nuke, de_dust2_fundo, de_dust2_so_fundo, cs_assault, de_airstrip, de_dust2_fundo, cs_italy, de_dust2, surf_megawave, de_dust2_fundo, aim_headshot, de_aztec, de_dust2, cs_assault, and de_dust2.

Game	Players	Map	Latency
Counter-Strike	3 / 16	de_dust2	46
Counter-Strike	9 / 10	de_dust2_fundo	781
Counter-Strike	0 / 12	de_nuke	39
Counter-Strike	6 / 6	de_dust2_fundo	42
Counter-Strike	2 / 2	de_dust2_fundo	42
Counter-Strike	3 / 14	de_dust2	42
Counter-Strike	8 / 10	de_dust2_so_fundo	42
Counter-Strike	17 / 18	de_dust2	42
Counter-Strike	3 / 24	cs_assault	55
Counter-Strike	0 / 16	de_airstrip	42
Counter-Strike	8 / 8	de_dust2_fundo	42
Counter-Strike	4 / 14	cs_italy	57
Counter-Strike	14 / 15	de_dust2	62
Counter-Strike	9 / 10	surf_megawave	62
Counter-Strike	10 / 10	de_dust2_fundo	62
Counter-Strike	4 / 24	aim_headshot	62
Counter-Strike	15 / 16	de_aztec	62
Counter-Strike	11 / 12	de_dust2	62
Counter-Strike	10 / 16	cs_assault	62
Counter-Strike	14 / 17	de_dust2	62
Counter-Strike	7 / 15	de_dust2_fundo	62

Figura 9 – Tela do *Stream* - Acesso aos servidores do *Counter Strike*

Fonte: Próprio autor

3.6 Analisador de Protocolo – Wireshark

O Wireshark é um programa de código aberto que analisa os pacotes nas redes IP. O Wireshark possui versões para Windows, Unix, Linux e MAC. Pode-se usar o analisador de protocolo para situações como:

- Detecção de problemas na rede;
- Análise de segurança de redes;
- Desenvolvimento de novos protocolos ou aplicações;
- Estudo do funcionamento de uma rede.

Os analisadores de protocolos podem ser instalados em um computador com algum tipo de interface conectada à rede de onde se capturam os dados.

O analisador de protocolos interage diretamente com o adaptador de rede. Dependendo do sistema operacional instalado na máquina, é necessário instalar *drivers* adicionais que permitam esta interação direta entre o programa e o adaptador de rede, sem a interferência do sistema operacional.

3.7 Uso de equipamento de coleta e caracterização de tráfego

Foi usado o equipamento Trafip para coleta e caracterização com as seguintes características:

- Caracterização de tráfego que possibilita estruturação de relatórios, visões e análises conforme as necessidades gerenciais;
- Diversidade de formas de coleta, o que permite sua utilização em diferentes topologias e sistemas;
- As informações são disponibilizadas de uma forma simples e clara;

Na figura 10, pode-se observar os tipos de coletas permitidas pelo equipamento Trafip que pode ser através de envio de fluxos (flows) ou de captura por espelhamento de tráfego através da funcionalidade *port-mirror*.

Figura 10 – Diagrama de funcionamento do Trafip (www.telcomanager.com)

Fonte: www.telcomanager.com

3.8 Estrutura de Roteamento

A estrutura da rede proposta é baseada em conceitos de roteamento mostrados a seguir:

- Ponto de Troca de tráfego centralizado;
- Troca de tráfego direta entre participantes;
- Ponto de Troca de tráfego híbrido;
- Pontos de acesso de alta velocidade (Ilha para o acesso);

3.8.1 Ponto de Troca de Tráfego na Internet (IXP)

No *backbone* das operadoras, geralmente os problemas ocorrem nas conexões pagas com outras operadoras, onde normalmente os links ficam congestionados devido a necessidade de redução de custo, provocando perdas de pacotes e atraso. A grande vantagem dos IXP(s) é que geralmente não são cobrados e normalmente não há gargalos, além disto, reduzem o número de *saltos* entre diversos provedores em cascata. Os IXP(s) podem também ser chamados de

NAP – *Network Access Point* ou PTT – *Ponto de Troca de Tráfego*. Na Figura 11 observa-se a conectividade da Internet.

Figura 11 – Conectividade da Internet- IPv4 Internet Topology (2008)
Fonte: IPv4 Internet Topology (2008)

Conforme Sousa (2004), desde o surgimento da Internet, existe uma grande preocupação com o congestionamento dos links entre os provedores. Para diminuí-lo, tornou-se fundamental a criação do IXP, ou seja, uma rede de alta velocidade, em que os roteadores podem conectar-se com a finalidade de troca de tráfego dos seus clientes.

Muitos provedores já utilizam o IXP. A utilização correta de um IXP permite ganhos de tempo de entrega pelo uso de caminhos alternativos (redundância) mais curtos entre as localidades, possibilitando uma economia significativa em links da Internet, além do ganho de desempenho.

Na Internet, um *Autonomous System* (AS) está, na maioria dos casos, associado a um provedor que fornece infra-estrutura de transmissão e acesso para uma região. Cada AS é reconhecido por um número decimal, utilizado pelo protocolo de roteamento BGP. O AS pode ser privado ou público. Caso seja público, deve ser devidamente registrado por um provedor em um órgão de registro.

O *Border Gateway Protocol* (BGP) é um protocolo de roteamento entre *Sistemas Autônomos* (Rekhter, 1995), usado para troca de rotas entre os participantes do IXP. Seu conhecimento é fundamental para a implementação e operação de um IXP.

O BGP no IXP é responsável pela política de roteamento aplicada, ou seja, gerenciar situações, tais como: (1) Um participante não quer importar para sua Tabela de roteamento todas as rotas aprendidas; (2) Um participante não quer anunciar todas as rotas da sua Tabela de roteamento para os outros participantes; (3) Um participante deseja modificar informações associadas à uma rota.

O controle do tráfego de entrada e saída do participante no IXP é realizado pela configuração do BGP no servidor de roteamento e no roteador do participante.

No Brasil, o primeiro IXP foi criado inicialmente pela FAPESP em São Paulo, posteriormente transferido para uma instituição privada. Outro ponto de troca é o IXP Metro que prove a infraestrutura para conexão de redes. Está mais destinado para as regiões metropolitanas (Belo Horizonte, Brasília, Curitiba, Florianópolis, Porto Alegre, Rio de Janeiro e São Paulo).

Na figura 12 pode-se observar o gráfico do tráfego no IXP Metro. Percebe-se que em um horário de pico o tráfego chega em até 7 Gigabit(s).

Figura 12 – Exemplo de utilização do IXP Metro- IXP Metro(2008)

Fonte: Metro – IXP Metro (2008)

A seguir, é apresentada a Figura 13 que representa os pontos de trocas de tráfego na Internet.

Figura 13 – IXP(S) distribuídos nos continentes (Packet Clearing House, 2009)

Fonte: Packet Clearing House (2009)

Segundo Feldman, (2007) uma estimativa do pacote ao percorrer a distância entre dois pontos é obtido através de algoritmos, pois seu percurso na Internet é aleatório, ou seja, a quantidade de saltos que o pacote percorrerá não é previsível. No caso de uma estrutura de troca de tráfego, o protocolo é dinâmico, porém, a distância entre os pontos são menores em números de saltos.

Pode-se observar no gráfico da Figura 14 a relação entre distância e atraso do pacote, na média a distância pode estar relacionada ao atraso, especialmente quando ocorre o aumento da quantidade de saltos e a passagem de links congestionados.

Figura 14 – Gráficos ilustrando relação de distância por atraso (Feldman, 2007)
Fonte: Feldman (2007)

3.8.2 Servidor de Roteamento

A troca de rotas da rede de um ponto de troca de tráfego pode ser executada por servidores de roteamento. Quando se faz uso de um servidor de roteamento, as políticas de troca de tráfego devem estar centralizadas neste servidor, tendo uma administração única. Os servidores de roteamento são computadores executando um *software* que realiza as funções de troca e processamento de rotas (Peno Filho, 2000).

O servidor de roteamento é composto de um servidor que executa um sistema operacional, na maioria das vezes *Linux* ou *Solaris*, e um módulo de roteamento, que pode ser livre ou proprietário. Dentre os livres, destacam-se o *Zebra*, *Gated* e o *Quagga*.

O servidor de roteamento recebe as rotas, processa de acordo com a política de roteamento implementada e, finalmente, as distribui para cada participante. Os servidores de roteamento usam o BGP como protocolo de roteamento.

Um fator muito importante é que o tráfego não passa no servidor de roteamento. Eles usam a capacidade do BGP de distribuir as rotas com o próximo destino, apontando para o roteador que anunciou a rota. Portanto, o tráfego é trocado diretamente entre os participantes (Figura 15).

Figura 15 – Troca de rotas no servidor de roteamento, Halabi e McPherson (2000)
Fonte: Halabi e McPherson (2000)

Os participantes de um IXP, que na maioria das vezes são provedores, têm como objetivo trocar tráfego de forma direta entre os participantes e de acordo com a política de roteamento implementada no IXP e nos participantes. Ao analisar a Figura 15, observa-se um IXP central que recebe as rotas de todos os participantes e divulga aquelas aprendidas. O servidor de roteamento não precisa ser uma máquina com muita capacidade de processamento pois o tráfego dos participantes não passa pelo servidor de roteamento; somente a Tabela de rotas trocadas.

Na Figura 16, observa-se um servidor de roteamento central com sessões BGP estabelecidas para cada participante. Através do protocolo BGP serão aprendidas e divulgadas as redes entre os participantes e o servidor de roteamento.

Figura 16 – Exemplo de troca de tráfego entre os participantes. Sousa (2004)

Fonte: Sousa (2004)

3.8.2.1 Servidor de Roteamento Zebra

O *Zebra*, quando instalado, funciona como um roteador, permitindo a troca de rotas com outros roteadores.

O *Zebra* suporta o protocolo BGP-4, assim como RIPv1, RIPv2 e OSPF. É original em seu projeto, pois tem um processo para cada protocolo, no qual cada módulo pode ser tratado independentemente do outro. Tem o seu código aberto para alterações e não é comercializado. No entanto, já existem empresas que fazem servidores de roteamento com base no *Zebra*, incluindo funções adicionais e ganhando os direitos de comercialização do serviço.

Os servidores de roteamento *Zebra* possuem uma sintaxe de configuração muito parecida com a linguagem similar ao fabricante Cisco. Como os equipamentos da Cisco são padrões de mercado, esta facilidade se estende para a maioria dos provedores.

3.8.3 Troca de tráfego direta entre participantes

No modelo de troca direta de tráfego entre os participantes é oferecido um ponto de conectividade de alta velocidade entre os provedores, neste caso não é usado o servidor de roteamento. A topologia física de conectividade pode ser provida por empresas desenvolvedoras de jogos. O administrador do IXP divulga os contatos dos participantes para a negociação dos anúncios e recebimento de redes. A escolha dos participantes pode ser feita através do estudo da intenção de tráfego dos jogadores para os servidores de jogos.

Os participantes, através do protocolo de roteamento BGP, controlam o anúncio e recebimento de suas redes diretamente para estes participantes. As redes são anunciadas com uma prioridade menor dentro de sua estrutura interna de roteamento, desta forma a redundância do ponto de troca pode ser a própria infraestrutura de rede da operadora. O participante pode sempre filtrar anúncios, independente da troca de tráfego.

3.8.4 Ponto de troca de tráfego híbrido

O Ponto de troca híbrido é uma combinação dos dois anteriores, ou seja, existe um servidor centralizado administrando as trocas de rotas e existem trocas de rotas independentes do servidor. O administrador do IXP divulga os contatos dos participantes, configura as políticas de roteamento para os interessados a trocar de tráfego.

3.8.5 Pontos de acesso de alta velocidade – Ilha para acesso

Os pontos de acesso de alta velocidade, basicamente são uma extensão da conectividade para “ilhas” de forma que usuários passam conectar-se diretamente nestas estruturas. As ilhas podem ser usadas para eventos que reúnem jogadores, pesquisas a conteúdos da redes e vídeo conferências específicas.

3.8.6 Escalabilidade do IXP

Switch

A escalabilidade no *switch* pode ser alcançada por meio do aumento do número de portas do *switch* ou da inserção de mais *switches* na estrutura do IXP.

Participante

Os links para a estrutura neste trabalho, estão previstos para 1 Gigabit, porém existe a possibilidade de migrar para interfaces de 10 Gigabit, ou fazer agregação chegando até 40 Gigabit.

Servidores de Jogos, aplicações

O limite de jogadores nos servidores pode ser tratado de modo independentemente da estrutura do IXP.

Servidor de roteamento

Como o tráfego do IXP não passa pelo servidor de roteamento, seu consumo de disco-memória é muito pequeno, permitindo a entrada de novos participantes sem afetar o desempenho do servidor. Os acordos de tráfego direto não farão uso de servidor de roteamento.

Endereço IP para os participantes

É recomendado estipular uma classe de endereço que permita um crescimento. Caso isso não seja feito, será necessário mudar o endereço de todos participantes para incluir uma nova classe.

3.9 Roteamento e Protocolo BGP

3.9.1 Conceitos de roteamento

Na Internet, o roteador é responsável por receber e encaminhar pacotes. O protocolo de roteamento usado na Internet é o BGP- Boeday Gateway Protocol. Cada roteador decide o melhor caminho, com base nas condições físicas da Internet informadas pelos protocolos de roteamento. (STALLINGS, 1997).

O roteador usa a Tabela de roteamento para escolher a melhor rota. A tabela de roteamento é formada pelas interfaces diretamente conectadas, rotas estáticas e protocolos de roteamento. A escolha da melhor rota para a tabela de roteamento é definida por pesos e métricas, na Tabela 2 observa-se o pesos dos protocolos usuais:

Tabela 2 – Pesos de rotas e protocolos de roteamento (DEETC, 2003)
Fonte: DEETC (2003)

	Cisco	Juniper M
Directamente ligada	0	0
IF Local	-	0
Rota estática	1	5
External BGP	20	170
EIGRP	90	-
IGRP	100	-
OSPF	110	
IS-IS	115	15, 18
RIP v1, v2	120	100
Internal BGP	200	170

A função do protocolo de roteamento é determinar a melhor rota para cada destino e distribuir as informações entre as redes, dentro do mesmo *Autonomous System*. Os protocolos de roteamento estão divididos em dois grupos: internos, Internal Gateway Protocol - IGP e os externos, External Gateway Protocol – EGP.

O anúncio das redes dentro de um sistema autônomo é feito por meio de um protocolo de roteamento interno, os protocolos de roteamento externo – EGP são responsáveis por descrever os caminhos disponíveis para transmissão do tráfego entre provedores diferentes.

Existem três famílias de protocolos de roteamento: o Distance Vector, o *Link State* e o Path Vector.

- **Distance Vector:** cada roteador recebe toda a Tabela de roteamento, escolhendo a melhor rota com base na distância de dois pontos, o Rip é um exemplo deste protocolo;
- **Link State:** cada roteador mantém uma topologia da rede, atualizada por meio de anúncios sobre os estados dos *links* e pontos de rede, o OSPF é um exemplo deste protocolo;
- **Path Vector:** cada roteador recebe informações com o caminho para chegar a cada rede. Por meio de atributos recebidos, pode-se definir o melhor caminho para cada destino, o BGP é um exemplo deste protocolo;

3.10 Sistema Autônomo – AS

Em redes de grande porte, o uso de um mesmo protocolo de roteamento em todos os roteadores não é viável devido ao elevado número de rotas.

O AS- Sistema Autônomo é um grupo de roteadores com uma administração técnica em comum. São administrados com uma política comum de roteamento. A Internet é composta por uma coleção de AS.

Uma solução é estruturar a rede de forma hierárquica, dividindo-a em vários sistemas autônomos. Exemplo desta utilização é a Internet.

Facilita a administração da rede e reduz a complexidade da escolha de rotas.

Em redes TCP_IP cada sistema autônomo é identificado por um número de 16 bits (Autonomous System Number – ASN).

O *Autonomous System* – AS, pode ser compreendido também pela definição de Nemeth (1995), como uma coleção de redes sob o controle de uma única autoridade central.

Na Internet, um *Autonomous System* – AS está, na maioria dos casos, associado a um provedor que fornece infra-estrutura de transmissão e acesso para uma região ou ainda uma empresa de grande porte com alta necessidade de endereços na Internet.

A administração e registro dos AS devem ser feitos na American Registry for Internet Numbers – ARIN.

A faixa de valores válidos para identificação de um AS varia de 1 a 65.535, sendo a subfaixa de 64.512 a 65.535 reservada para uso privado.

3.11 Protocolo de roteamento BGP v4

O protocolo BGP é usado para troca de rotas na Internet e entre os participantes do IXP. Seu conhecimento é fundamental para a implementação e operação de um IXP e para administração de anúncios em um sistema autônomo.

O BGP é um protocolo de roteamento entre *Autonomous Systems* (Rekhter, 1995), que teve sua versão original datada de 1989, com o BGP-1. Posteriormente, em 1993, o BGP-4 começou a ser implementado.

Conforme Parkhurst (2001) o BGP é um protocolo de roteamento robusto e escalável. O BGP é um protocolo estável quando bem administrado, ou seja, necessita de bons técnicos para a sua administração. Como evidência há o fato de o BGP ser o protocolo usado na Internet. Atualmente, a Internet tem em sua Tabela de rotas BGP mais de 150.000 anúncios.

O protocolo BGP está especificado nas RFCs 1771, 1772, 1773, 1774, 1863, 1930, 1965, 1966, 1997, 1998, 2042, 2283, 2385, 2439.

Conforme visto anteriormente, o BGP é um protocolo baseado no algoritmo Distance Vector que transmite informações de prefixos.

Na camada de transporte o protocolo BGP usa o porta 179, devido a este fato, o transporte das mensagens não precisa ser feito pelo BGP. O BGP suporta CIDR (Classless Inter-Domain Routing).

O BGP no IXP é responsável pela política de roteamento aplicada, ou seja, gerenciar situações, tais como:

- Um participante não quer importar para sua Tabela de roteamento todas as rotas aprendidas, ou seja filtros de entrada;
- Um participante não quer anunciar todas as rotas da sua Tabela de roteamento para os outros participantes, uso de filtros de saída;
- Um participante deseja modificar informações associadas a uma rota, filtros de mudanças de atributos;

O controle do tráfego de entrada e saída do participante no IXP é realizado pela configuração do BGP no servidor de roteamento e no roteador do participante.

Dois roteadores que formam uma conexão TCP com a finalidade de trocar rotas, são chamados de *neighbors*, ou seja, vizinhos. O BGP troca informações de roteamento entre os vizinhos e não precisam estar diretamente conectados.

Figura 17 – Roteadores vizinhos do BGP
Fonte: Próprio autor

Na Figura 17, dois roteadores estabelecem uma seção BGP. Observa-se que cada roteador tem um endereço IP e ambos podem pertencer ao mesmo AS ou AS diferentes.

Os vizinhos do BGP estabelecem seções que podem ser External BGP (EBGP), no caso vizinhos de AS diferentes, ou Internal BGP (IBGP), se eles forem vizinhos do mesmo AS, na Figura 18 observa-se exemplos de vizinhos IBGP e EBGP.

Na estrutura proposta os participantes poderão ter como vizinhos o servidor de roteamento, caso a troca seja central ou com outras operadoras se for troca direta ou ainda se for com ambos os casos, seja híbrida (uso de servidores de roteamento e operadoras). Estas conexões com vizinhos podem receber redes iguais, porém com prioridades distintas.

Figura 18 – Exemplos de IBGP e E-BGP
Fonte: Próprio autor

O detalhamento do funcionamento do protocolo BGP tais como comandos utilizados na configuração e análise podem ser encontrados no APÊNDICE J.

4 CARACTERIZAÇÃO E SIMULAÇÃO DO JOGO NA INTERNET E NA REDE PROPOSTA

4.1 Caracterização do Tráfego

A caracterização de tráfego do Counter Strike foi realizada para viabilizar a simulação, pois era necessário conhecer as características do tráfego do cliente e servidor para posteriormente simular no NS2.

Inicialmente foi usado o Wireshark para coletar o tráfego do Counter Strike. Para tratamento dos dados coletados do Wireshark usou-se o Phytion. Houve uma certa dificuldade de caracterização do tráfego usando o Wireshark pois era necessário filtrar e ordenar as sessões através do Phytion. Desta forma usou-se o equipamento Trafip.

O equipamento Trafip trabalha com os módulos apresentados na Figura 19:

Figura 19 – Módulos do equipamento Trafip

Fonte: www.telcomanager.com

1- Colector- É responsável por coletar dados. A coleta pode ser feita de várias formas, como Netflow, Sflow, SNMP e Port Mirror. No caso deste trabalho usou-se o Port Mirror, pois o switch da LAN House não suportava as demais funcionalidades.

2- Summarizer- Responsável por tratar os dados para posteriormente armazenar em banco de dados. O tratamento é baseado em agrupamento de objetos. Os objetos podem ser IP, portas TCP/UDP, Subredes.

3- Reporter- Geração de relatórios e tabelas para análise. Vide relatório do Apêndice F.

4- Database- Organização e armazenamento de dados

4.2 Coleta de dados

A coleta foi realizada em uma LAN House, com 4 jogos simultaneamente. Foram instalados dois jogos em cada servidor. Foram usadas duas versões distintas de Counter Strike em momentos de coletas diferentes (das 22:00 até as 6:00 na primeira versão e a segunda versão das 10:00 até as 19:00). A amostra foi significativa, pois durante este período havia até 40 usuários gerando-se um tráfego de até três megabit por segundo atingindo até 5000 pacotes por segundo.

Na Figura 20 observa-se a tela de administração dos servidores através do programa denominado HLTV usado para gerenciamento. Este programa mede atraso através do teste de conectividade até os servidores, monitora também os usuários em cada servidor.

Figura 20 – Tela do programa HLTV
Fonte: Próprio autor

Durante a coleta observou o tráfego de pacotes no Trafip. Nos gráficos das Figuras 21 e 22 pode-se observar a quantidade coletada em bits por segundo e pacotes respectivamente:

Figura 21 – Tráfego coletado para caracterização de tráfego- bits por segundo

Fonte: Próprio autor

Figura 22 – Pacotes coletados para caracterização de tráfego - pacotes por segundo

Fonte: Próprio autor

4.3 Caracterização do tráfego na LAN House

O jogo Counter Strike foi instalado em dois servidores com endereços IP(s) e portas respectivamente (192.168.0.23, 192.168.0.42), portas (27015 e 27016). Os demais endereços IP(s) iniciando com 192.168 são clientes acessando o servidor de jogo.

Na análise com 100 fluxos e um período de coleta de aproximadamente 18 horas (0:00- 18:00 horas), caracterizou-se o tráfego através do equipamento Traflp.

Os jogadores tiveram todos os recursos disponíveis sem serem atrapalhados por atrasos, visto que estavam conectados através de uma rede local. O atraso nesta etapa não era fator importante, o objetivo foi coletar os tamanhos dos pacotes enviados entre os clientes e servidores para posteriormente inserir as informações no simulador de rede.

As informações geradas se encontram no APENDICE F. A seguir na Tabela 3 apresenta-se uma planilha ilustrando um relatório de coleta:

Tabela 3 – Acessos nos servidores de jogos

Fonte: Próprio autor

Source IP	Destination IP	Source port	Destination port	Bytes	Packets	Flows	Taxa (Kbps)
192.168.0.23	192.168.0.25	27015	27005	98,102,147	933,114	34	36.334
192.168.0.23	192.168.0.33	27015	27005	92,864,421	819,908	31	34.394
192.168.0.23	192.168.0.2	27015	27005	84,780,369	774,007	30	31.400
192.168.0.23	192.168.0.24	27015	27005	77,789,595	676,283	26	28.811
192.168.0.23	192.168.0.26	27015	27005	74,917,718	634,878	26	27.747
192.168.0.23	192.168.0.29	27015	27005	72,804,887	635,924	30	26.965
192.168.0.42	192.168.0.17	27016	27005	71,806,464	638,646	24	26.595
192.168.0.23	192.168.0.30	27015	27005	71,637,468	598,107	29	26.532
192.168.0.23	192.168.0.32	27015	27005	71,186,489	612,575	27	26.365
192.168.0.23	192.168.0.241	27015	27005	69,849,120	601,950	28	25.870
192.168.0.42	192.168.0.20	27016	27005	69,730,719	628,424	26	25.826
192.168.0.42	192.168.0.16	27016	27005	68,810,569	621,967	23	25.485
192.168.0.42	192.168.0.19	27016	27005	67,300,322	569,360	23	24.926
192.168.0.42	192.168.0.18	27016	27005	67,167,844	575,980	22	24.877
192.168.0.42	192.168.0.2	27016	27005	63,291,820	542,140	20	23.441
192.168.0.33	192.168.0.23	27005	27015	60,312,224	827,699	31	22.338
192.168.0.25	192.168.0.23	27005	27015	60,276,932	822,515	34	22.325
192.168.0.42	192.168.0.33	27016	27005	59,129,605	510,539	19	21.900
192.168.0.42	192.168.0.25	27016	27005	58,505,472	487,393	18	21.669
192.168.0.42	192.168.0.26	27016	27005	57,245,085	490,014	18	21.202
192.168.0.42	192.168.0.24	27016	27005	53,748,792	457,115	18	19.907
192.168.0.24	192.168.0.23	27005	27015	50,286,529	684,691	26	18.625
192.168.0.23	192.168.0.111	27015	27005	49,725,368	405,627	29	18.417
192.168.0.2	192.168.0.23	27005	27015	49,540,986	685,992	30	18.349
192.168.0.23	192.168.0.12	27015	27005	49,197,348	401,918	16	18.221
192.168.0.42	192.168.0.111	27016	27005	47,739,827	383,960	19	17.681
192.168.0.23	192.168.0.14	27015	27005	47,526,416	389,345	15	17.602

A partir dos dados obtidos calculou-se a média do tamanho dos pacotes, desvio padrão e variância do cliente para servidor e servidor para cliente.

A seguir, a Tabela 4 ilustrando os pacotes coletados do servidor para cliente:

Tabela 4 – Consolidação da caracterização servidor cliente

Fonte: Próprio autor

Servidor	Soma dos tamanhos dos pacotes (bytes)	Número de Pacotes	Tamanho Médio pacotes (bytes)
192.168.0.23	98102147	933114	105,1341497
192.168.0.23	92864421	819908	113,2620013
192.168.0.23	84780369	774007	109,5343698
192.168.0.23	77789595	676283	115,0252113
192.168.0.23	74917718	634878	118,0033298
192.168.0.23	72804887	635924	114,4867736
192.168.0.42	71806464	638646	112,4354713
192.168.0.23	71637468	598107	119,7736659
192.168.0.23	71186489	612575	116,2086096
192.168.0.23	69849120	601950	116,0380763
192.168.0.42	69730719	628424	110,9612602
192.168.0.42	68810569	621967	110,6337941
192.168.0.42	67300322	569360	118,20346
192.168.0.42	67167844	575980	116,6148894
192.168.0.42	63291820	542140	116,7444203
192.168.0.42	51047181	449266	113,6235126
192.168.0.42	59129605	510539	115,8179982
192.168.0.42	58505472	487393	120,0375713
192.168.0.42	57245085	490014	116,8233663
192.168.0.42	53748792	457115	117,5826477
192.168.0.23	49725368	405627	122,5889006
192.168.0.23	49197348	401918	122,4064312
192.168.0.42	47739827	383960	124,3354178
192.168.0.23	47526416	389345	122,0676161
192.168.0.23	47296158	398733	118,6161115
192.168.0.23	47108849	393199	119,809178
192.168.0.42	47046695	385884	121,9192685
192.168.0.23	46516082	403008	115,4222298
192.168.0.23	46336587	393112	117,8712097
192.168.0.23	45646581	407410	112,0408949
192.168.0.23	45398934	393717	115,308544
192.168.0.42	42792582	351735	121,6614269
192.168.0.23	41984071	370069	113,4493054

(continua)

192.168.0.42	41658865	349496	119,1969722
192.168.0.42	39941384	333926	119,6114828
192.168.0.42	142127477	1386713	102,4923521
192.168.0.42	137170320	1284300	106,8055127
192.168.0.23	134824479	1301972	103,5540542
192.168.0.42	131429220	1226001	107,2015602
192.168.0.42	130535670	1226414	106,4368721
192.168.0.42	119575878	1091313	109,570653
192.168.0.23	108403023	1046379	103,5982402
192.168.0.23	105703653	1034639	102,1647676
192.168.0.23	100854689	892259	113,0329747
192.168.0.23	98900673	1001521	98,75047353
192.168.0.23	98545442	889106	110,8365504
192.168.0.23	97158897	864465	112,3919384
192.168.0.23	95097808	828736	114,7504247
192.168.0.23	93849301	832956	112,6701783
192.168.0.23	93643350	838533	111,6752114
192.168.0.23	93206310	853604	109,1915103
192.168.0.23	92647029	833230	111,1902224
192.168.0.23	91945968	837591	109,7743027
192.168.0.23	91310125	830182	109,9880809
192.168.0.23	89576282	798380	112,1975525
192.168.0.23	80896018	759565	106,503088
192.168.0.23	80010878	759590	105,3342961
192.168.0.23	79546970	759754	104,7009558
192.168.0.23	60824844	522235	116,4702557
192.168.0.42	60518289	596682	101,4246936
192.168.0.23	60293657	521721	115,5668585
192.168.0.23	59212270	511563	115,7477574
192.168.0.23	58975725	506763	116,3773302
192.168.0.23	58764307	505375	116,2786188
192.168.0.42	55730673	465097	119,8259137
192.168.0.42	54966237	505516	108,7329323
		Média	113,3103894 bytes
		Variância	34,83082456
		Desvio Padrão	5,901764529

Pode-se observar na Tabela 5 os pacotes coletados do cliente para servidor:

Tabela 5 – Consolidação da caracterização cliente servidor

Fonte: Próprio autor

Cliente	Soma dos tamanhos dos Pacotes	Número de Pacotes	Tamanho Médio pacotes
192.168.0.17	74904368	1252049	59,82542856
192.168.0.32	69321885	696813	99,48420164
192.168.0.18	68526469	1154212	59,37078197
192.168.0.19	65276965	1079759	60,45512471
192.168.0.20	63756746	1065992	59,80977906
192.168.0.32	63621671	1067794	59,58234547
192.168.0.16	62666836	1049594	59,70578719
192.168.0.33	55559468	960053	57,87125086
192.168.0.241	55184414	936583	58,92100753
192.168.0.111	54514141	893871	60,98658643
192.168.0.29	53993538	916317	58,92451848
192.168.0.34	49886546	822466	60,65484287
192.168.0.24	49414643	836204	59,09400457
192.168.0.25	48991106	816579	59,99554973
192.168.0.37	48829853	797442	61,23310912
192.168.0.38	47266704	768480	61,50674578
192.168.0.41	46964923	770587	60,94694434
192.168.0.2	46227949	768515	60,15230542
192.168.0.33	60312224	827699	72,86733946
192.168.0.25	60276932	822515	73,28368723
192.168.0.24	50286529	684691	73,44412151
192.168.0.2	49540986	685992	72,21802295
192.168.0.26	47256221	637167	74,16614639
192.168.0.17	46407009	629962	73,66636242
192.168.0.20	45363905	637966	71,10708878
192.168.0.241	45310111	611368	74,11266373
192.168.0.29	44958621	615192	73,08063336
192.168.0.30	44467339	601702	73,90259464

(continua)

192.168.0.16	43829762	608029	72,08498608
192.168.0.32	43185821	580959	74,33540233
192.168.0.18	43051606	589546	73,02501586
192.168.0.19	42621618	579280	73,5768851
192.168.0.2	39462473	548682	71,9223029
192.168.0.33	38948073	522896	74,48531448
		Média	67,34702591 bytes
		Variância	77,04114015
		Desvio Padrão	8,777308252

Comparando-se as Tabelas 4 e 5, vê-se que o tamanho do pacote do servidor para o cliente é maior do que do cliente para o servidor.

A média utilizada na simulação foi 67,35 bytes com desvio padrão de 8,78 bytes do cliente para o servidor e média de 113,31 bytes do servidor para o cliente com desvio padrão de 5,9 bytes. Estes valores foram coletados no experimento de caracterização de tráfego conforme apresentado anteriormente.

Conforme (Feldman, 2007) a média dos pacotes do cliente para o servidor do Counter Strike é 74,32 bytes e do servidor para o cliente é 121,99 bytes, valores próximos dos coletados na LAN House.

4.4 Fases da Simulação

Conforme observado na Figura 23, a caixa (cenário Counter Strike) no diagrama de simulação é representado os tráfegos gerados através da disponibilização de servidores Counter Strikes e/ou uso de Lan Houses. Posteriormente o tráfego foi analisado pelo Wireshark e TRAFIP representados neste mesmo diagrama como (logs de Rede).

Observa-se no diagrama da Figura 23 como (Analisador do Perfil de Rede) o resultado da caracterização do tráfego, permitindo a implementação do tráfego do cliente e servidor do Counter Strike no simulador NS2.

Para a simulação foi necessário descrever o cenário, isso pode ser feito na linguagem TCL interpretada pelo simulador de rede NS-2. Foi implementado um programa gerador para facilitar a confecção dos arquivos de TCL(s), visto na caixa (Gerador de Cenários NS2) no diagrama da Figura 23.

Posteriormente, a simulação foi executada utilizando-se o NS-2 com os parâmetros da rede proposta.

São gerados dois arquivos, um do tipo NAM e um outro de “trace”. Através do arquivo tipo NAM é possível visualizar a simulação. Com o uso de um programa analisador, visto na caixa (Analisador de tráfego da Simulação) do diagrama da Figura 23, são obtidas informações do arquivo de “trace” para gerar estatísticas e gráficos que permitam fazer comparações e analisar os possíveis ganhos.

A simulação foi feita procurando-se usar o cenário mais próximo do tráfego coletado; neste momento valida-se a simulação.

Figura 23 – Diagrama de fases da simulação

Fonte: Próprio autor

4.5 Experimentos com o Simulador NS-2

A fim de estudar a redução de atraso com o uso da rede proposta, baseada em IXP conectadas paralelamente à Internet, utilizou-se o simulador de redes NS-2.

No NS-2 serão implementados três cenários. Todos os cenários utilizarão o perfil do tráfego do Counter Strike.

Conforme apresentado na metodologia, a geração dos cenários foi feita através de arquivos do tipo TCL que descreve o ambiente a ser analisado com informações de topologias, quantidade de hosts e links, caracterizando toda a rede.

Existem diversos geradores de topologias para o NS-2, porém não foi encontrado algo que atendesse à necessidade desta tese, desta forma foi elaborado um programa em C capaz de gerar arquivos TCL, denominado gerador. Este arquivo encontra-se documentado no apêndice A.

A geração dos arquivos em TCL a partir de um programa permitiu a execução de várias simulações desejadas de forma prática e automática conseqüentemente, fazer um grande número de simulações: complexas e inviáveis de se escrever sem o auxílio de um programa, que pode ser alterado para Trabalhos Futuros , descrito no item 6.2.

O NS-2 consegue gerar arquivos (denominados arquivos de “trace”) com informações sobre os pacotes trocados na rede, que possibilitam a criação de diversos gráficos e estatísticas para analisar a simulação.

Para simular o cenário adotou-se bandas escolhidas aleatoriamente com distribuição esperada, mostrada na Tabela 6. Para atraso entre os roteadores (quando nenhum dos extremos é um IXP) admitiu-se valores escolhidos aleatoriamente com distribuição uniforme no intervalo de 5 a 20 milisegundos. Esta medida foi adotada com base em diversos traces para servidores na Internet.

Tabela 6 – Bandas alocadas na simulação**Fonte:** Próprio autor

Banda (Mega Bits por segundo)	Porcentagem de ocorrência
1 Mb	20%
2 Mb	20%
34 Mb	20%
156 Mb	30%
1 Gb	10%

Os valores médios de links, atrasos e taxa de congestionamento foram atribuídos após pesquisas efetuadas em uma Operadora de Telecomunicação, e análise de traces de rotas na Internet.

Conforme mencionado, a média do tamanho de pacote e desvio padrão foram obtidos através dos experimentos com tráfego Trafip e estes foram utilizados como parâmetros das distribuições que serão mencionadas a seguir.

Conforme o (LANG,2003) a distribuição lognormal é a que mais se adequa ao tamanho dos pacotes do servidor para o cliente dos jogos do estilo FPS, enquanto os pacotes do cliente para o servidor seguem uma distribuição normal. Na simulação foram adotados estes valores para o tamanho dos pacotes, ou seja, do servidor para cliente adotou-se a distribuição lognormal enquanto que para os pacotes de cliente para servidor utilizou-se a distribuição normal.

Duas aplicações foram criadas e adicionadas ao NS-2 para gerenciar os tráfegos de cliente para servidor e de servidor para cliente. As aplicações foram nomeadas como CounterStrikeClient.cc e CounterStrikeServer.cc comentadas no Apêndice B e C respectivamente.

O roteamento usado foi o padrão do NS2, o SPF(Shortest Path First) baseado no algoritmo de Dijkstra. Este é o algoritmo usado no protocolo OSPF- Open Short Path First.

Este algoritmo é executado no início da simulação, onde as rotas são aplicadas de acordo com uma matriz com custos de links e topologias.

Este exemplo está muito próximo da realidade, pois geralmente as rotas internas das Operadoras são aprendidas e anunciadas pelo protocolo interno OSPF e posteriormente divulgadas pelo protocolo BGP.

4.6 Cenário Internet sem IXP

No primeiro cenário, o tráfego dos usuários não passará exclusivamente por IXP(s), na Internet tem diversos links e IXP(s) conectados.

O programa gerador apresenta um menu para receber a entrada das variáveis de acordo com a Figura 24, a seguir:

```
marxi@saruman: ~/Desktop/Trabalho
File Edit View Terminal Tabs Help
.marxi@saruman:~/Desktop/Trabalho$ ./gerador
Que tipo de cenário você deseja simular?
1 - Internet sem IXP.
2 - IXP.
3 - IXnet somente com IXP.
1
Digite o número de servidores.
3
Digite o número de clientes para o 1o.servidor.
4
Digite o número de clientes para o 2o.servidor.
5
Digite o número de clientes para o 3o.servidor.
6
Digite o número de roteadores (nem clientes nem servidores) na rede.
30
Digite um fator de congestionamento da rede entre 0 e 1.
0 - Sem congestionamento
1 - Totalmente congestionada (impossível utilizar)
0.5
Digite a proporção esperada de ligações entre os roteadores.
0.2
Digite o nome do arquivo TCL a ser gerado.
cenariol.tcl
marxi@saruman:~/Desktop/Trabalho$
```

Figura 24 – Execução do programa analisador

Fonte: Próprio autor

Neste menu, pode-se escolher os cenários, números de servidores, clientes para cada servidor, número de roteadores, proporção esperada de conexões (no caso do cenário 1), quantidade de switches (no caso do cenário 3) e índice de congestionamento dos links.

Foi implementado um programa que analisa o arquivo de “trace”, tratando o tamanho de pacotes, atrasos, número de saltos e outras informações necessárias para análise. Este programa escrito em C foi nomeado como analisador.c e está comentado no apêndice D.

Os programas implementados foram utilizados para os três cenários, pode-se alterar os programas desenvolvidos para simular outro tipo de aplicação. Observa-se na Figura 25, um diagrama ilustrando o uso dos programas:

Figura 25 – Diagrama dos programas implementados
Fonte: Próprio autor

Pode-se observar no diagrama que o gerador, poderá gerar códigos em TCL de acordo com o cenário escolhido. O cenário é escolhido através da definição das variáveis tais como número de servidores, usuários e roteadores.

O NS-2 junto com as aplicações adicionais citadas, simula o tráfego do cliente e servidor. Finalmente, o programa analisador dos resultados é utilizado para gerar os gráficos e estatísticas relevantes.

Para simular o congestionamento é permitido escolher na implementação do cenário um fator de congestionamento entre 0 e 1 que representa uma diminuição na proporção de utilização da banda e um aumento de atraso. Desta forma, um fator igual à 0,4 significa que a banda deverá diminuir em quarenta por cento do esperado

No primeiro cenário serão realizados três experimentos, o primeiro somente para validação do tamanho do pacote, o segundo experimento para simulação da Internet e o terceiro experimento para simular a Internet com um fator de congestionamento.

O gerador inicia a topologia conectando os roteadores aleatoriamente com base nas variações dos parâmetros fornecidos no menu inicial, posteriormente aloca servidores e usuários aleatoriamente.

A seguir, para analisar o tamanho dos pacotes foi criado um cenário relativamente pequeno, com os parâmetros da Tabela 7.

Tabela 7 – Variáveis do primeiro experimento

Fonte: Próprio autor

Servidores	Usuários
Quantidade de Servidores	3
Servidor 1	10
Servidor 2	15
Servidor 3	20
Roteadores	50
Congestionamento	0,5
Proporção de ligação	0,05 (-índice baixo)
Tempo simulação	4 segundos

A partir desta simulação, gerou-se a topologia a seguir na Figura 27, no NS-2 foi configurada a linha vermelha para identificar o tráfego do servidor para o cliente enquanto a azul do cliente para o servidor.

Figura 27 – Topologia 1, cenário 1

Fonte: Próprio autor

Pode-se observar no gráfico da Figura 28, a média de tamanho de pacotes por tempo para esta simulação.

Figura 28 – Tamanho de pacotes Cliente Servidor / Servidor Cliente

Fonte: Próprio autor

Como previsto na caracterização de tráfego executada na prática, percebe-se que o tamanho do pacote simulado está na faixa dos dados obtidos na caracterização do tráfego. Para melhor visualização, pode-se observar nas Figuras 29 e 30 o tráfego individual do cliente para servidor e servidor para o cliente.

Figura 29 – Tamanho do pacote do Cliente para Servidor
Fonte: Próprio autor

A simulação realizada tem 45 usuários distribuídos em 3 servidores, passando por 50 roteadores, pode-se observar na figura 29 que a quantidade de pacotes dos clientes para o servidor é maior do que servidor para cliente.

Figura 30 – Tamanho do pacote do Servidor para o cliente
Fonte: Próprio autor

Pode-se observar na figura 30 que o tamanho do pacote do servidor inicia em um valor mais alto que o do cliente e seu comportamento é mais constate.

Conforme Lang, (2003) um servidor de Half Life envia um pacote para cada cliente em média a cada 60 milisegundos enquanto o cliente envia para o servidor em 50 ou 33 milisegundos, o comportamento do Half Life é muito próximo do Counter Strike, desta forma foi adotada esta medida nesta simulação, acarretando ainda mais pacotes no sentido do cliente para o servidor.

Para estudar um cenário com mais usuários, executou-se uma simulação alterando os parâmetros de acordo com a Tabela 8:

Tabela 8 – Variáveis do segundo experimento do primeiro cenário

Fonte: Próprio autor

Servidores	Usuários
Quantidade de Servidores	10
Servidor 1	20
Servidor 2	15
Servidor 3	5
Servidor 4	10
Servidor 5	20
Servidor 6	30
Servidor 7	5
Servidor 8	10
Servidor 9	15
Servidor 10	20
Roteadores	100
Congestionamento	0,4
Proporção de ligação	0,03
Tempo simulação	10 segundos

A seguir, na Figura 31, a topologia gerada para este cenário:

Figura 31 – Topologia experimento 2 – cenário 1

Fonte: Próprio autor

Pode-se observar no pacote vermelho, o tráfego do servidor para o cliente e o azul do cliente para o servidor.

Na Figura 32, vê-se uma tela do programa analisador disponibilizando estatísticas sobre o tamanho de pacotes, atraso e saltos além de gerar arquivos com variáveis para implementar os gráficos.

```

marxi@saruman: ~/Desktop/Trabalho
File Edit View Terminal Tabs Help
marxi@saruman:~/Desktop/Trabalho$ ./analizador
Digite o nome do arquivo de trace a ser analisado.

^C
marxi@saruman:~/Desktop/Trabalho$ cp entrada entrada2
marxi@saruman:~/Desktop/Trabalho$ gedit entrada2
marxi@saruman:~/Desktop/Trabalho$ ./analizador <entrada2
Digite o nome do arquivo de trace a ser analisado.
Digite o número de servidores.
Digite o número de clientes para o 1o.servidor.
Digite o número de clientes para o 2o.servidor.
Digite o número de clientes para o 3o.servidor.
Digite o número de clientes para o 4o.servidor.
Digite o número de clientes para o 5o.servidor.
Digite o número de clientes para o 6o.servidor.
Digite o número de clientes para o 7o.servidor.
Digite o número de clientes para o 8o.servidor.
Digite o número de clientes para o 9o.servidor.
Digite o número de clientes para o 10o.servidor.
Digite o número de roteadores (nem clientes nem servidores) na rede.
Media do tamanho dos pacotes S->C: 112.69808
Media do tamanho dos pacotes C->S: 66.88370

Media do delay dos pacotes S->C: 120.99487
Media do delay dos pacotes C->S: 112.58387

Media de saltos dos pacotes S->C: 6.70667
Media do saltos dos pacotes C->S: 6.70667
marxi@saruman:~/Desktop/Trabalho$

```

Figura 32 – Tela do programa analisador

Fonte: Próprio autor

Podem-se observar os atrasos como 120,99 ms do servidor para o cliente e 112,58 ms do cliente para o servidor. Se considerar a somatória do atraso, obterá 233,57 ms, um atraso considerável que atrapalharia o desenvolvimento de um jogo ou uma outra aplicação sensível.

Como base de comparação na Tabela 9, seguem os dados fornecidos pelo Internet Traffic Report, considerando também o tempo de ida e volta.

Região	Média de tempo (ms)
Ásia	295
Austrália	184
Europa	226
America do Norte	90
America do Sul	125

Tabela 9 – Média de tempo de resposta por continente (internettrafficreport.com, 2009)

Fonte: internettrafficreport.com (2009)

Esta medida é coletada de cinco em cinco segundos a partir de um ponto central na América do Norte e medindo o tempo de resposta de vários continentes. Os valores de simulação estão próximos do cenário atual, validando-se a taxa de atraso.

A seguir, nas Figuras 33 e 34, apresenta-se o gráfico do atraso pela quantidade de pacotes, percebe-se também que o atraso na simulação é praticamente constante visto que durante o intervalo de tempo da simulação não ocorre mudanças de topologias.

Figura 33 – Atraso pela quantidade de pacotes Cliente-Servidor

Fonte: Próprio autor

Figura 34 – Atraso pela quantidade de pacotes Servidor-Cliente
Fonte: Próprio autor

Pode-se observar nas Figuras 35 e 36 principalmente a relação (de proporcionalidade inversa) do número de saltos por atraso.

Figura 35 – Atraso pela quantidade de pacotes e saltos Cliente Servidor
Fonte: Próprio autor

Conforme a característica desta simulação, tanto o número de usuários (clientes) na simulação quanto o intervalo de saída dos pacotes dos servidores influenciaram no fato de haver mais pacotes no sentido do cliente para o servidor, observado na Figura 35.

Figura 36 – Atraso pela quantidade de pacotes e saltos Servidor-Cliente
Fonte: Próprio autor

No tráfego do servidor-cliente, conforme Figura 36, observa-se que tanto o número de saltos quanto o atraso parecido, mesmo sendo pacotes um pouco maiores e mais constantes como visto anteriormente.

O gráfico de barras da Figura 37 mostra os atrasos do ponto de vista de cada usuário. Os usuários estão dispostos no eixo das coordenadas e para cada um estão representadas duas barras, uma vermelha para o atraso médio dos pacotes do servidor que está conectando até ele e uma verde para o atraso médio dos pacotes no sentido contrário.

Figura 37 – Atraso pela quantidade de pacotes Cliente Servidor
Fonte: Próprio autor

A tese trata especialmente da variação do atraso gerado por links congestionados e aumento de saltos (roteadores), para isto foi implementado um terceiro experimento com objetivo de simular um fator de congestionamento de 70 % dos links. Em pesquisas com uma Operadora, foi informado que este valor de congestionamento ocorre em algumas situações, especialmente nas de redundância. Foi realizada a simulação com os mesmos dados da Tabela 8, obtendo-se o seguinte resultado:

Média do tamanho dos pacotes S->C: 112,69 bytes

Média do tamanho dos pacotes C->S: 66,88 bytes

Média do atraso dos pacotes S->C: 187,60 ms

Média do atraso dos pacotes C->S: 171,42 ms

Média de saltos dos pacotes S->C: 7,53

Média do saltos dos pacotes C->S: 7,53

A seguir nas Figuras 38 e 39 uma comparação entre a Internet sem congestionamento e a Internet congestionada através do tráfego de pacotes do servidor para o cliente. Observa-se que na Internet com o fator de congestionamento em setenta por cento, o atraso e número de saltos são significativos, nos piores casos, atrasos acima de 500 ms (somatória do cliente servidor para o servidor cliente) a Internet torna-se inviável para aplicações sensíveis a atraso.

Figura 38 – Comparação Cenário 1- Internet com e sem Congestionamento Cliente-Servidor
Fonte: Próprio autor

Na Figura 39, observa-se o gráfico análogo com os pacotes no sentido contrário.

Figura 39 – Comparação Cenário 1- Internet com e sem Congestionamento Servidor-Cliente
Fonte: Próprio autor

4.8 Segundo Cenário - Simulação da Internet com um IXP

O segundo cenário, mostra que o tráfego será passado exclusivamente por um IXP(s), conforme a Figura 40.

Figura 40 – Exemplo ilustrativo de um acesso pela Internet usando um IXP
Fonte: Próprio autor

O switch central foi configurado com banda de um Gigabit por segundo para os participantes e atraso de 5 milissegundos para cada conexão.

A topologia foi implementada de forma que o usuário e o servidor estarão no máximo 3 saltos do switch central

A simulação foi feita com a Tabela 10 a seguir:

Servidores	Usuários
Quantidade de Servidores	3
Servidor 1	15
Servidor 2	10
Servidor 3	5
Roteadores	15
Congestionamento	0,4
Tempo simulação	10 segundos

Tabela 10 – Variáveis do primeiro experimento do segundo cenário
Fonte: Próprio autor

Na Figura 41 pode-se observar a topologia gerada:

Figura 41 – Topologia experimento 2 – cenário 2

Fonte: Próprio autor

Análise para este cenário constatou-se:

Média do tamanho dos pacotes S->C: 112,69 bytes

Média do tamanho dos pacotes C->S: 66,88 bytes

Média do atraso dos pacotes S->C: 68,63 ms

Média do atraso dos pacotes C->S: 66,95 ms

Média de saltos dos pacotes S->C: 4,86

Média de saltos dos pacotes C->S: 4,86

Pode-se observar na Figura 42, o gráfico que mostra o comportamento da rede na simulação de um IXP, observa-se que o maior atraso entre o Cliente Servidor ou servidor cliente está no máximo em 110 milissegundos e o número de saltos no máximo de seis.

Figura 42 – Atraso pela quantidade de pacotes e saltos Cliente Servidor IXP simulado
Fonte: Próprio autor

4.9 Terceiro Cenário – Simulação da rede proposta – IXnet

Este cenário ilustrado na Figura 43, é a proposta desta tese, ou seja, uma Internet alternativa com conexão à Internet atual, formada a partir de IXP(s) conectados por fibras de 1 Gíabit por segundo. Nesta rede somente tráfegos de aplicações exigentes a atrasos serão roteados.

Figura 43 – Exemplo ilustrativo de acessos do cenário 3

Fonte: Próprio autor

O gerador no menu inicial pergunta quantos IXP(s) terão, e posteriormente divide os roteadores nas suas vizinhanças e a partir daí conecta os IXPs aleatoriamente (com um resultado seguramente conexo garantido novamente pelo uso do Union-Find) com banda de um Gigabit por segundo para os participantes e atraso de 5 milissegundos para cada conexão, deixando os clientes e os servidores em uma distância de até 3 saltos até o IXP mais próximo.

Foi simulado usando-se os mesmos dados da Tabela 8 do cenário 1, usando 10 IXP(S), vê-se na Figura 44 o diagrama de simulação no NS2.

Figura 44 – Topologia experimento cenário 3

Fonte: Próprio autor

Após a simulação executou-se o programa analisador para geração das médias para análise e os arquivos para os gráficos. Resultando em:

Média do tamanho dos pacotes S->C: 112,69 bytes

Média do tamanho dos pacotes C->S: 66,88 bytes

Média do atraso dos pacotes S->C: 69,47 ms

Média do atraso dos pacotes C->S: 63,67 ms

Média de saltos dos pacotes S->C: 6,67

Média de saltos dos pacotes C->S: 6,67

4.10 Implementando Protocolo BGP

Apesar da Internet e os IXP(s) usarem o protocolo BGP, não foi o foco deste trabalho explorar o protocolo para anúncio e recebimento de rotas e sim ver se atenderia uma possível simulação de roteamento, sugerida nos trabalhos futuros.

O gerador de cenários foi desenvolvido para atender as topologias de tamanhos variáveis em grande escala. O seu código não foi adaptado para suportar o BGP, porém foi implementado um cenário mais simples apenas para exemplificar os diferentes tipos de tráfegos do Counter Strike e do BGP.

Para configurar o protocolo BGP no simulador de rede NS2 deve-se carregar o módulo BGP.

Inicialmente configurou-se o BGP nos roteadores, posteriormente estabeleceu-se a vizinhança e anunciou-se as redes.

O BGP controla a entrada e saída de rotas com base nas configurações em cada sistema autônomo.

Esta simulação foi baseada na alteração de um exemplo de reflexão de rotas do BGP proposta por (Feng, 2003). Basicamente, existem três Sistemas Autônomos com onze nós. Pode-se inicialmente notar os pacotes pretos que são usados para estabelecimento da sessão do BGP. Posteriormente, após o estabelecimento da vizinhança inicia-se o tráfego do Counter Strike do servidor para o cliente e cliente para Servidor, eventualmente, durante a simulação trafegam pacotes BGP para confirmação se o vizinho está ativo.

Foi implementado somente dois agentes UDP, um deles para exemplificar o tráfego de cliente para servidor e outro do servidor para o cliente. Foram feitas também configurações de EBGP e IBGP e conceitos de roteamento utilizados e estão no ANEXO J.

Segue na Figura 45, os possíveis pacotes: azul do cliente para o servidor, vermelho do servidor para o cliente e o preto do BGP.

Figura 45 – Topologia experimento BGP
Fonte: Próprio autor

Complementando, na Figura 46 segue exemplo da Tabela de roteamento usada pelo BGP.

```

BGP routing table of n1
BGP table version is 10, local router ID is 10.0.1.1
Status codes: * valid, > best, i - internal.
  Network        Next Hop        Metric LocPrf Weight Path
*> 10.1.10.0/24  10.0.2.1/32      -     -     - 1    i
*> 10.2.9.0/24   10.0.7.1/32      -     -     - 2    i

BGP routing table of n2
BGP table version is 2, local router ID is 10.0.2.1
Status codes: * valid, > best, i - internal.
  Network        Next Hop        Metric LocPrf Weight Path
*> 10.1.10.0/24  10.1.8.1/32      -     -     - 1
*> 10.2.9.0/24   10.0.7.1/32      -     -     - 2    i

BGP routing table of n3
BGP table version is 2, local router ID is 10.0.3.1
Status codes: * valid, > best, i - internal.
  Network        Next Hop        Metric LocPrf Weight Path
*> 10.1.10.0/24  10.0.2.1/32      -     -     - 1    i
*> 10.2.9.0/24   10.0.7.1/32      -     -     - 2    i

BGP routing table of n4
BGP table version is 2, local router ID is 10.0.4.1
Status codes: * valid, > best, i - internal.
  Network        Next Hop        Metric LocPrf Weight Path
*> 10.1.10.0/24  10.0.2.1/32      -     -     - 1    i
*> 10.2.9.0/24   10.0.7.1/32      -     -     - 2    i

BGP routing table of n5
BGP table version is 9, local router ID is 10.0.5.1

```

Figura 46 – Tabela de roteamento do protocolo BGP
Fonte: Próprio autor

Pode-se observar na Figura 46 os roteadores recebendo as redes com o próximo destino (Next Hop).

5 ANÁLISE DE RESULTADOS

Inicialmente, na análise dos resultados comparou-se o cenário da Internet sem congestionamento com o cenário proposto. Nesta comparação, foi incluída a mesma entrada de dados, observa-se no gráfico da figura 47 a comparação em saltos, atrasos e pacotes da Internet e a IXnet.

Figura 47 – Comparação da Internet / IXnet do tráfego cliente para servidor
Fonte: Próprio autor

Pode-se observar nas Figuras 48 e 49 que a rede IXnet é menor em saltos e atrasos para o mesmo tipo de aplicação.

Figura 48 – Comparação da Internet / Ixnet do tráfego servidor para cliente
Fonte: Próprio autor

Do ponto de vista do usuário, pode-se observar na Figura 49 em relação ao atraso, verifica-se a diferença quanto ao atraso para cada aplicação.

Figura 49 – Comparação de atraso da Internet / Ixnet do tráfego por usuário
Fonte: Próprio autor

Adicionalmente, para obter uma comparação mais genérica, executou-se várias vezes, simulações dos cenários 1 e 3 alterando o número de roteadores. Nesta variação foram mantidos 3 servidores e 30 clientes. O índice de congestionamento foi mantido igual a 0,4. Os parâmetros de simulação encontram-se nas Tabelas 11 e 12.

A proporção esperada de ligações de cada simulação do cenário 1 foi normalizada em $2/n$, onde n é o número de roteadores. Isto foi feito para que o número de ligações esperadas sejam da ordem do número de roteadores e não do seu quadrado, tornando mais fiel a comparação entre cada simulação.

Tabela 11 – Amostra para simulação Cenário 1

Fonte: Próprio autor

Dados da Simulação	Resultado da Análise Cenário 1
Servidor:3 Clientes:30 Roteadores:50	Média do atraso dos pacotes S->C: 130,73 ms Desvio padrão do atraso dos pacotes S->C: 39,11 ms Média do atraso dos pacotes C->S: 121,53 ms Desvio padrão do atraso dos pacotes C->S: 37,40ms Média de saltos dos pacotes : 6,52
Servidor:3 Clientes:30 Roteadores:100	Média do atraso dos pacotes S->C: 156,11 ms Desvio padrão do atraso dos pacotes S->C: 48,48 ms Média do atraso dos pacotes C->S: 133,01 ms Desvio padrão do atraso dos pacotes C->S: 44,61 ms Média de saltos dos pacotes : 7,43
Servidor:3 Clientes:30 Roteadores:150	Média do atraso dos pacotes S->C: 172,24 ms Desvio padrão do atraso dos pacotes S->C: 59,83 ms Média do atraso dos pacotes C->S: 163,56 ms Desvio padrão do atraso dos pacotes C->S: 58,78 ms Média de saltos dos pacotes: 9,45
Servidor:3 Clientes:30 Roteadores:200	Média do atraso dos pacotes S->C: 208,13 ms Desvio padrão do atraso dos pacotes S->C: 73,38 ms Média do atraso dos pacotes C->S: 195,32 ms Desvio padrão do atraso dos pacotes C->S: 67,83 ms Média de saltos dos pacotes: 11,35

(continua)

Servidor:3 Clientes:30 Roteadores:250	Média do atraso dos pacotes S->C: 213,12 ms Desvio padrão do atraso dos pacotes S->C: 51,34 ms Média do atraso dos pacotes C->S: 194,61 ms Desvio padrão do atraso dos pacotes C->S: 47,75 ms Média de saltos dos pacotes: 11,08
Servidor:3 Clientes:30 Roteadores:300	Média do atraso dos pacotes S->C: 225,78 ms Desvio padrão do atraso dos pacotes S->C: 61,17 ms Média do atraso dos pacotes C->S: 211,79 ms Desvio padrão do atraso dos pacotes C->S: 59,62 ms Média de saltos dos pacotes : 11,65
Servidor:3 Clientes:30 Roteadores:350	Média do atraso dos pacotes S->C: 250,28 ms Desvio padrão do atraso dos pacotes S->C: 93,91 ms Média do atraso dos pacotes C->S: 225,90 ms Desvio padrão do atraso dos pacotes C->S: 80,40 ms Média de saltos dos pacotes : 11,63

Para as simulações correspondentes no terceiro cenário (Ixnet) adotou-se o número de switches para cada simulação como $3n/50$ onde n é o número de roteadores, esta escolha segue de uma média de 16 roteadores para cada IXP.

Tabela 12 – Amostra para simulação Cenário 3

Fonte: Próprio autor

Dados da Simulação	Resultado da Análise Cenário 3
Servidor:3 Clientes:30 Roteadores:50	Médiado atraso dos pacotes S->C: 57,34 ms Desvio padrão do atraso dos pacotes S->C: 14,89 ms Média do atraso dos pacotes C->S: 53,10 ms Desvio padrão do atraso dos pacotes C->S: 13,25 ms Média de saltos dos pacotes : 5,23

(continua)

Servidor:3 Clientes:30 Roteadores:100	Média do atraso dos pacotes S->C: 69,19 ms Desvio padrão do atraso dos pacotes S->C: 22,34 ms Média do atraso dos pacotes C->S: 65,16 ms Desvio padrão do atraso dos pacotes C->S: 17,96 ms Média de saltos dos pacotes : 6,03
Servidor:3 Clientes:30 Roteadores:150	Média do atraso dos pacotes S->C: 77,60 ms Desvio padrão do atraso dos pacotes S->C: 15,56 ms Média do atraso dos pacotes C->S: 66,44 ms Desvio padrão do atraso dos pacotes C->S: 16,22 ms Média de saltos dos pacotes : 6,00
Servidor:3 Clientes:30 Roteadores:200	Médiado atraso dos pacotes S->C: 79,36 ms Desvio padrão do atraso dos pacotes S->C: 19,52 ms Médiado atraso dos pacotes C->S: 78,79 ms Desvio padrão do atraso dos pacotes C->S: 19,48 ms Média de saltos dos pacotes: 8,72
Servidor:3 Clientes:30 Roteadores:250	Médiado atraso dos pacotes S->C: 89,08 ms Desvio padrão do atraso dos pacotes S->C: 23,06 ms Médiado atraso dos pacotes C->S: 74,33 ms Desvio padrão do atraso dos pacotes C->S: 14,20 ms Médiade saltos dos pacotes : 7,56
Servidor:3 Clientes:30 Roteadores:300	Médiado atraso dos pacotes S->C: 83,59 ms Desvio padrão do atraso dos pacotes S->C: 27,10 ms Médiado atraso dos pacotes C->S: 79,34 ms Desvio padrão do atraso dos pacotes C->S: 23,60 ms Médiade saltos dos pacotes : 7,74
Servidor:3 Clientes:30 Roteadores:350	Médiado atraso dos pacotes S->C: 88,78 ms Desvio padrão do atraso dos pacotes S->C: 18,94 ms Médiado atraso dos pacotes C->S: 88,17 ms Desvio padrão do atraso dos pacotes C->S: 18,82 ms Médiade saltos dos pacotes : 7,67

A seguir nas Figuras 50 e 51 apresenta-se os resultados obtidos na simulação com os dados das Tabelas 11 e 12. Pode-se observar a variação de roteadores e seus respectivos atrasos e saltos.

Figura 50 – Comparação geral da Internet / Ixnet do tráfego cliente para servidor

Fonte: Próprio autor

Figura 51 – Comparação geral da Internet / Ixnet do tráfego servidor para cliente
Fonte: Próprio autor

6 CONCLUSÃO

A distância de saltos entre o cliente e o servidor, o congestionamento dos links e a aleatoriedade do caminho do pacote na Internet podem causar atraso em aplicações sensíveis a esta variável. Baseando-se nos fatores supracitados, esta tese propõe uma Internet especializada, somente para aplicações sensíveis a atraso, denominada aqui como IXnet.

No desenvolvimento da tese analisou-se, com o uso do simulador de rede NS2, o tráfego do jogo Counter Strike, aplicação sensível a atraso escolhida como referência neste trabalho. A simulação exigiu a caracterização do tráfego feita com uso de analisador de rede e equipamento para coleta e análise.

A caracterização do tráfego permitiu conhecer as características de tráfego do Counter Strike e executar uma simulação com características próximas do ambiente real. O processo de caracterização utilizado nesta tese pode ser adaptado para novos tipos de tráfegos tais como voz, vídeo entre outros.

Na simulação foram gerados quatro programas, o programa gerador para confeccionar os arquivos usados no simulador de acordo com variáveis fornecidas (servidores, roteadores, usuários, congestionamento), duas aplicações para simular o tráfego do jogo, (cliente para o servidor e outra do servidor para o cliente) e por fim um programa para analisar o resultado da simulação e gerar dados para os gráficos.

Os programas gerados neste trabalho podem ser adaptados para complementar outras funcionalidades (protocolos) bem como testar outras características, tais como aumento de bandas, atrasos, roteadores e servidores.

A rede proposta é formada através da conexão de pontos de trocas de tráfego com links de alta velocidade, exclusivo para aplicações sensíveis a atraso foi desenvolvida a partir da realização de experimentos de IXP (Apêndice I) e simulações em NS-2.

A melhora no tempo de resposta é obtida através da redução do número de saltos, um congestionamento menor (controlado) e pelo aumento dos enlaces e equipamentos nas centrais da IXnet.

A análise dos resultados comprovou que a redução do atraso e saltos do cenário proposto é significativa, conforme comparação das Figuras 50 e 51.

Como conclusão, diante de todos os estudos e análises realizadas durante o desenvolvimento do trabalho, a IXnet mostra-se possível e indicada como uma alternativa para uma Internet especializada mais controlada e menos congestionada destinada à aplicações sensíveis a atraso.

6.1 Recomendação IXNET

A Ixnet pode ser implementada conectada a Internet, facilitando migrações das aplicações. O tráfego sensível a atraso pode ser roteado através do anúncios de redes com uma prioridade melhor dentro de cada sistema autônomo.

Pode-se usar o conceito de ilhas de acessos, para centrais conectadas diretamente nos IXP mais próximos. A idéia da IXNet é aproximar cada vez mais o usuário final dos servidores, ou seja, o Datacenter com a Operadora. A centralização da administração dos IXP(s) facilitará o gerenciamento do tráfego, o congestionamento e rastreamento de eventos de segurança. Recomenda-se também uma integração entre os administradores dos IXP(s) usando-se marcação em rotas para rapidamente bloquear um tráfego não desejado, esta marcação pode ser feita através do protocolo BGP.

Atualmente na Internet tem-se dificuldade da integração entre sistemas diferentes, no tempo adequado, para resolução de um evento de segurança.

O projeto para implementação da IXnet pode ser iniciado através de conexões novas entre IXP(S) existentes e listados no ANEXO A.

Tanto a conexão entre os IXPs quanto seus equipamentos centrais sejam eles switches ou roteadores podem ser bem dimensionados, visto que são centrais. Este trabalho, não usou conexões 10 Gigabits por segundo, porém basta alterar o programa e executar mais simulações.

A Figura 52 ilustra a IXnet, apesar da Internet não estar representada, cada sistema autônomo pertencente à IXnet pode ter sua conexão à Internet passando todo o tipo de tráfego e a IXnet destinada para tráfegos selecionados. Caso ocorra uma queda na IXnet pode-se transferir o tráfego pela Internet, mesmo que com isso possa haver mais lentidão.

Figura 52 – Ilustrativa - Tendência IXnet
Fonte: Próprio autor

6.2 Trabalhos Futuros

Adaptar os programas implementados nesta tese para outros tipos de jogos FPS ou aplicações de voz e vídeo e videoconferência, comparando analiticamente os resultados;

Adaptar os programas para execução de novas funcionalidades tais como QOS, MPLS para priorização de tráfego e redução de atraso;

Explorar o protocolo BGP no simulador NS2 para tratar as possíveis convergências causadas por quedas de links na Internet;

Estudar e analisar novos ganhos da rede proposta, especialmente em segurança e gerenciamento.

7 REFERÊNCIAS

ANIDO, R.; PENHA, A. **Infra-estrutura para jogos distribuídos**. In: DAUSHA, R. M. et al. WORKSHOP DE PARCERIA – UNICAMP & SIEMENS; (2.: 2004: Campinas, SP) Caderno de resumos do Workshop de Parceria – UNICAMP & Siemens – Fase II, 18 de junho de 2004: Agência de Inovação - UNICAMP: Siemens, p. 13-15, 2004.

BREU, L., **Online Games: Traffic Analysis of Popular Game Servers (Counter Strike: Source)**. Tese TIK - Institut für Technische Informatik und Kommunikationsnetze. Zurich, 2007.

BUCHLEIT, R.F. **Atraso Compensation in Networked Computer Games**, 2004. 23p. (Tese). Disponível em: < http://vorlon.case.edu/~vxl11/NetBots/rfb_report.pdf> Acesso em: 05 jul. 20008.

CLAYPOOL, M.; LAPOINT, D.; WINSLOW, J. **Network Analysis of Counter Strike and Starcraft**. In: *PROC. OF THE 22ND IEEE INTERNATIONAL PERFORMANCE, COMPUTING, AND COMMUNICATIONS CONFERENCE (IPCCC)*. p. 263-270, 2003.

CHIZZOTTI, Antônio. **Pesquisa em Ciências Humanas e Sociais**. 5.ed. São Paulo: Cortez, 2001.

DEETC - Departamento de Engenharia Elétrica e Telecomunicações e de Computadores - **Protocolos de encaminhamento**, Disponível em: <http://http://www.deetc.isel.ipl.pt/redesdecomunic/disciplinas/RC2/acetatos/TCP-IP5a%20Routing.pdf>, Lisboa, 2003.

FÄRBER, J. **"Network Game Traffic Modelling"**, *Proceedings of NetGames 2002*, pp. 74--78, Braunschweig (BRD), 16--17 April 2002.

FELDMAN, D.; SHAVITT, Y. **An Optimal Median Calculation Algorithm for Esimating InternetLink Delays From Active Measurements**. In: END-TO-END MONITORING TECHNIQUES AND SERVICES, 2007. E2EMON'07 Workshop Disponível em: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4261339.

FENG, W.C.; CHANG, F.; FENG, W.C.; WALPOLE, J. **Provisioning On-line Games: A Traffic Analysis of a Busy Counter Strike Server**. In: THE INTERNET MEASUREMENT WORKSHOP. p. 151-156. Marseille, 2002.

GARDENGHI, L.; PIFFERI, S.; D'ANGELO, G.; BONONI, L. **Design and evaluation of a migration - based architecture for massively populated Internet Games**. In: THE FIRST IEEE INTERNATIONAL WORKSHOP ON NETWORKING ISSUES IN MULTIMEDIA ENTERTAINMENT. GLOBECOM, p. 166-175. Bologna, Italy, 2004.

GUNBOUND. **Horus Spirit**. Disponível em:<<http://www.gbound.com.br>>. Acesso em: 05 jul. 20008.

HENDERSON, T.; BHATTI, S. **Networked games – a QoS-sensitive application for QoS-insensitive users?** In: PROCEEDINGS OF THE ACM SIGCOMM WORKSHOP ON REVISITING IP QOS, p. 141-147. Disponível em: <http://www.cs.st-andrews.ac.uk/tristan/pubs/ripqos03.pdf>. Karlsruhe, Germany, 2003.

HALABI, S.; McPHERSON, D. **Internet Routing: Architectures. 2ed. Networking Technology: Cisco Press**, 480p. 2000.

Internet System **Consortium**. Disponível em <[http:// www.isc.org](http://www.isc.org)>. Acesso em: 05 jul. 2008.

Internet Traffic **Report**, disponível em: <http://internettrafficreport.com> Acesso em 25 de março de 2009.

IPv4 Internet **Topology**. Disponível em:<<http://www.caida.org/home/>>. Acesso em: 05 jul. 2008.

IXP **Metro**. Disponível em:<<http://IXP.br/cgi-bin/all>>. Acesso em: 05 jul. 2008.

JOHARI, R.; TSITSIKIKLIS, J. **Network Resource Allocation and a Congestion Game**. In: THE 42ND IEEE CONFERENCE ON DECISION AND CONTROL, 2003. p. 2112-2117. Disponível em: 05 jul. 2008.
<http://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=28482&isYear=2003>.

JUNIPER, C. K. **BGP v4 Tutorial**. São Paulo, abril 2003.

LANG, T; ARMITAGE, G; BRANCH, P; YICHOO, **HNetworked games – a A Synthetic Traffic Model for Half-Life - Centre for Advanced Internet Architectures**, 2003, Centre for Advanced Internet Architectures. Disponível em: <http://atnac2003.atcrc.com/ORALS/Lang2.pdf>.

LAPOINTE, D., WINSLOW, J.; **Analyzing and Simulating Network Game Traffic**. In: A MAJOR QUALIFYING PROJECT REPORT SUBMITTED TO THE FACULTY OF THE WORCESTER POLYTECHNIC INSTITUTE in partial fulfillment of the Degree of Bachelor of Science, 2001.

MARTINS, A. G. **Manual para elaboração de monografias e dissertações**. 3^a ed.: São Paulo - SP: Ed. Atlas, 2002.

MATTAR, F. G. **Pesquisa de marketing**. Ed Compacta. São Paulo: Atlas, 1996.

MMOGCHART. Disponível em:<<http://www.mmogchart.com/>>. Acesso em: 04 nov. 2008.

MULTIREDE INFORMÁTICA LTDA. **BGP Configuração e Depuração**. São Paulo, 2000.

MULTIREDE INFORMÁTICA LTDA. **QOSIP- Qualidade de Serviço sobre IP**. São Paulo, 2007.

Packet Clearing House, Disponível em <<http://www.pch.net/>>. Acesso em: 22 mar. 2009.

PARKHURST, W. Cisco BGP-4 **Command and Configuration Handbook**. Indianapolis - USA: Cisco Press, 2001.

PENO FILHO, R. **Operação e Administração de IXPs no Brasil**. São Paulo: Comitê Gestor da Internet no Brasil, 2000.

Redes Comunitárias de educação e pesquisa, **COMEP** Disponível em <<http://www.redecomep.rnp.br/>>. Acesso em: 22 mar. 2009.

REKHTER, Y. RFC 1771: **A Border Gateway Protocol BGP-4**, T. J. Watson **Research Center. IBM Corp, T. Li, Cisco System**, 1995. Disponível em: <<http://www.ietf.org/rfc/rfc1771.txt>>.

SHAIKH, A.; SAHU, S.; ROSU, M.; SHEA, M.; SAHA, D. **Implementation of a service Platform for online games**. In: THE 2ND WORKSHOP ON NETWORK AND SYSTEM SUPPORT FOR GAMES, 2004. Disponível em: <http://portal.acm.org/citation.cfm?id=963917>.

SILVA, C. E.; ARRAES D.; PEDROSA, D.; TRINTA, F.; WANDERLEY I.; PITA, M.; PORTELA M. MACHADO, M.; BORGES, R.; RAMALHO, G. **Uma plataforma para jogos móveis massivamente multiusuários**. In: ANAIS WJOGOS 2005 IV WORKSHOP BRASILEIRO DE JOGOS E ENTRETENIMENTO DIGITAL, 2005, p. 300.

SOUSA, A. J. B. **Implementação de um IXP para comutar tráfego entre sistemas autônomos**. Dissertação (Mestrado em Engenharia de Computação), Instituto de Pesquisa de São Paulo – USP., São Paulo, 2004.

STALLINGS, W. **Local & Metropolitan Area Network**. 1^a ed.: Upper Saddle River, New Jersey; Prentice Hall, Inc, 1997.

Telcomanager, **Diagrama**, Disponível em <http://www.telcomanager.com>
Acesso em 25 de março de 2009.

ZHANG, HAISHU. **The effect of atraso on network games**. Tese. 32p. UMEA University, 2006.

ZHOU, Q.; MILLER, C.J.; BASSILIOUS, V. **First Person Shooter Multiplayer Game Traffic Analysis**. In: 11TH IEEE SYMPOSIUM ON OBJECT ORIENTED REAL - TIME DISTRIBUTED COMPUTING (ISORC), p. 195-200, 2008.

APÊNDICE A – Programa Gerador

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>

#define MAX_SERV 100 /* Número máximo de servidores */
#define MAX_NODES 10000 /* Número máximo da soma do número de clientes, de
servidores e de roteadores */
#define MAX_LINKS 4 /* Número máximo de conexões por nó no cenário 1 */

#define print( X... )    fprintf( arq_saida, X )

double banda();
void inicializaUF(int pai[MAX_NODES], int sz[MAX_NODES],int n);
int find(int pai[MAX_NODES], int x);
void unionUF(int pai[MAX_NODES], int sz[MAX_NODES], int n, int x, int y);

int main(){
    int v_serv[MAX_SERV]; /* Vetor que associa cada servidor ao seu número
de clientes */
    int n_serv, n_rot,n_clientes=0, n_nos=0, n_switches; /* Variáveis que
guardam a quantidade de cada tipo de nó */
    int i,j,k; /* Variáveis auxiliares */
    char nome_arq_saida [50];
    FILE * arq_saida;
    double p; /* Proporção esperada de ligações entre roteadores usada no
cenário 1 */
    double fator_cong; /* Fator de congestionamento */
    int opcao;
    int n_links[MAX_NODES]; /* Associa cada roteador ao número de ligações
em que ele está envolvido no cenário 1 */
    int vizinho, inicial, quantos; /* Variáveis auxiliares para a definiçã
das topologia */
    int pai[MAX_NODES],sz[MAX_NODES]; /* Vetores utilizados pelo Union-Find
*/

    srand(time(NULL));
    printf("Que tipo de cenário você deseja simular?\n");
    printf("1 - Internet sem IXP.\n");
    printf("2 - IXP.\n");
    printf("3 - IXnet somente com IXP.\n");
    scanf("%d", &opcao);
    printf("Digite o número de servidores.\n");
    scanf("%d",&n_serv);
    n_nos += n_serv;
    for(i=0;i<n_serv;i++){
        printf("Digite o número de clientes para o %do.servidor.\n",i+1);
        scanf("%d",&v_serv[i]);
        n_clientes += v_serv[i];
        n_nos += v_serv[i];
    }
    printf("Digite o número de roteadores (nem clientes nem servidores) na
rede.\n");
    scanf("%d",&n_rot);
    n_nos += n_rot;
    if(n_nos>MAX_NODES || n_serv >MAX_SERV){
        printf("Muitos nós ou muitos servidores. Finalizando gerador.\n");
```

```

        return 0;
    }
    printf("Digite um fator de congestionamento da rede entre 0 e 1.\n0 -
Sem congestionamento \n1 - Totalmente congestionada (impossível
utilizar)\n");
    scanf("%lf",&fator_cong);
    if(opcao == 1){
        printf("Digite a proporção esperada de ligações entre os
roteadores.\n");
        scanf("%lf",&p);
        /*p = p /(1.0 *n_rot);*/ /* Linha a ser utilizada para normalizar a
proporção como citado na tese */
    }
    else if(opcao == 3){
        printf("Digite a quantidade de switches.\n");
        scanf("%d",&n_switches);
    }
    printf("Digite o nome do arquivo TCL a ser gerado.\n");
    scanf("%s",nome_arq_saida);
    arq_saida = fopen(nome_arq_saida,"w");

    for(i=0;i<n_nos;i++)
        n_links[i]=0;
    inicializaUF(pai,sz,n_nos);

    /* Gerando cabeçalho no arquivo TCL */
    print("#Criando o simulador.\n");
    print("set ns [new Simulator]\n\n");
    print("#Definindo diferentes cores para os fluxos de dados.\n");
    print("$ns color 1 Blue\n");
    print("$ns color 2 Red\n\n");
    print("#Abrindo o arquivo de trace do NS.\n");
    print("set f [open out.tr w]\n");
    print("$ns trace-all $f\n\n");
    print("#Abrindo o arquivo de trace para o NAM.\n");
    print("set nf [open out.nam w]\n");
    print("$ns namtrace-all $nf\n\n");
    print("#Definindo uma rotina de finalizacao.\n");
    print("proc finish {} {\n");
    print("\tglobal ns nf f\n");
    print("\t$ns flush-trace\n");
    print("\t#Fechando o arquivo de trace do NS.\n");
    print("\tclose $f\n");
    print("\t#Fechando o arquivo de trace para o NAM.\n");
    print("\tclose $nf\n");
    print("\t#Executando o NAM no arquivo de trace.\n");
    print("\texec nam out.nam &\n\texit 0\n}\n\n");

    /* Criando nós... */
    print("# n_nos eh o numero de nos (contando clientes e
servidores).\n");
    print("set n_nos %i\n", n_nos );
    print("for {set i 0} {$i < $n_nos} {incr i} {\n" );
    print("\tset n$i [$ns node]\n}\n\n");

    /* Criação da topologia conectando os roteadores convenientemente */
    if(opcao==1){ /* Primeiro Cenário : Internet sem IXP */
        /* Gerando topologia aleatória. */
        for(i=n_serv+n_clientes;i<n_nos-1;i++){
            for(j=i+1;j<n_nos;j++){

```

```

        /* Os roteadores são conectados de forma aleatória segundo
a proporção esperada */
        if(n_links[i]<MAX_LINKS && n_links[j] <MAX_LINKS &&
((double)rand())/((double)RAND_MAX) <= p){
            print("$ns duplex-link $n%i $n%i %.3fMb %.0fms
DropTail\n",i,j,(1.0-fator_cong)*banda(),(1.0+fator_cong)*(rand()%16 + 5));
            unionUF(pai,sz,n_nos,i,j);
            n_links[i]++;
            n_links[j]++;
        }
    }
    /* Para manter o grafo conexo colocamos estas arestas
complementares. Elas são escolhidas
    analisando-se o Union-Find */
    print("\n#Arestas complementares \n");
    for(j=n_serv+n_clientes+1;j<n_nos;j++){
        if(find(pai,j)!=find(pai,n_serv+n_clientes)){
            print("$ns duplex-link $n%i $n%i %.3fMb %.0fms
DropTail\n",n_serv+n_clientes + rand()%(j- n_serv - n_clientes) ,j,(1.0-
fator_cong)*banda(),(1.0+fator_cong)*(rand()%16 + 5));
            unionUF(pai,sz,n_nos,j-1,j);
        }
    }
    print("\n");
}
else if(opcao==2){ /* Segundo cenário: IXP */
    /* Separado um roteador para ser o IXP, toma-se metade dos outros
para estarem diretamente ligados a ele */
    for(i = n_serv+ n_clientes + 1; i < n_serv+n_clientes + 1 + (n_rot-
1)/2 ; i++){
        /* As conexões que partem de um switch tem banda e atraso fixos
como definidos na tese. */
        print("$ns duplex-link $n%i $n%i 1000Mb 5ms
DropTail\n",i,n_serv+n_clientes);
    }
    /* A outra metade desses roteadores é conectada a roteadores do
primeiro tipo */
    for(i = n_serv+n_clientes + 1 + (n_rot-1)/2 ; i<n_nos;i++){
        vizinho = n_serv+n_clientes + 1 + rand()%(n_rot-1)/2;
        print("$ns duplex-link $n%i $n%i %.3fMb %.0fms
DropTail\n",i,vizinho,(1.0-fator_cong)*banda(),(1.0+fator_cong)*(rand()%16
+ 5));
    }
}
else{ /* Terceiro Cenário: IXnet*/
    /* decide-se quantos roteadores estarão associados a cada switch e
guarda-se na variável quantos */
    quantos = (n_rot - n_switches)/n_switches;
    for(j=0;j<n_switches;j++){ /* Para cada switch... */
        inicial = n_serv+ n_clientes + n_switches + j*quantos;
        /* ... separa-se metade dos seus roteadores associados e os
liga ao switch diretamente...*/
        for(i = inicial; i <= inicial +quantos/2 ; i++){
            print("$ns duplex-link $n%i $n%i 1000Mb 5ms
DropTail\n",i,n_serv+n_clientes+j);
        }
        /* ...deixando que os roteadores do outra metade se liguem aos
da primeira */
        for(i = inicial + quantos/2 + 1;(j==n_switches-1 && i<n_nos) ||
i < inicial + quantos;i++){
            vizinho = inicial +rand()%(quantos/2+1);

```

```

        print("$ns duplex-link $n%i $n%i %.3fMb %.0fms
DropTail\n",i,vizinho,(1.0-fator_cong)*banda(),(1.0+fator_cong)*(rand()%16
+ 5));
    }
}
/* Os switches devem ser interconectados com conexões velozes.
   A proporção dessas ligações se baseia no número de switches */
for(i=n_serv +n_clientes;i<n_serv+n_clientes + n_switches-1;i++)
    for(j = i+1;j<n_serv+n_clientes+n_switches;j++){
        if(((double)rand())/((double)RAND_MAX) <=
(2.0)/((double)n_switches)){
            print("$ns duplex-link $n%i $n%i 1000Mb 5ms
DropTail\n",i,j);
            unionUF(pai,sz,n_nos,i,j);
        }
    }
/* Caso o Union-Find mostre que a rede não está conexa arestas
entre switches são adicionadas para esse fim */
for(j=n_serv+n_clientes+1;j<n_serv + n_clientes+n_switches;j++)
    if(find(pai,j)!=find(pai,n_serv+n_clientes)){
        if(j!= n_serv + n_clientes + 1)
            print("$ns duplex-link $n%i $n%i 1000Mb 5ms
DropTail\n",n_serv+n_clientes + rand()%(j- n_serv-n_clientes-1) ,j);
        else
            print("$ns duplex-link $n%i $n%i 1000Mb 5ms
DropTail\n",n_serv+n_clientes,j);
            unionUF(pai,sz,n_nos,j-1,j);
    }
}
print("\n");

/* Conectando aleatoriamente servidores e clientes à rede. Para
qualquer um dos cenários. */
for(i=0;i<n_serv+n_clientes;i++){
    vizinho = rand()%n_rot +n_serv+n_clientes;
    print("$ns duplex-link $n%i $n%i %.3fMb %.0fms DropTail\n",i,
vizinho,(1.0-fator_cong)*banda(),(1.0+fator_cong)*(rand()%16 + 5));
}
print("\n");

for(i= n_serv ; i<n_serv + n_clientes;i++){
    /* Criando agentes para a transmissao dos pacotes cliente-
>servidor. */
    print("set udp%d [new Agent/UDP]\n",i);
    print("$udp%d set class_ 1\n",i);
    print("$ns attach-agent $n%d $udp%d\n\n",i,i);
    print("set cliente%d [new
Application/Traffic/CounterStrikeClient]\n",i);
    print("$cliente%d attach-agent $udp%d\n\n",i,i);
}

for(i=0;i<n_serv;i++){
    /* Criando agentes nulos (traffic sink).*/
    print("set null%d [new Agent/Null]\n",i);
    print("$ns attach-agent $n%d $null%d\n\n",i,i);
}

/* Conecta-se cada agente UDP de transmissão dos pacotes cliente-
>servidor ao traffic sink do servidor*/
j = n_serv;
for(i=0;i<n_serv;i++){

```

```

        for(k=0;k<v_serv[i];k++)
            print("$ns connect $udp%d $null%d\n",j++,i);
    }
    print("\n");

    k=n_serv;
    for(i=0 ; i<n_serv;i++){
        for(j=0;j<v_serv[i];j++){
            /* Criando agentes para a transmissao dos pacotes servidor-
>cliente. */
            print("set udp%d_%d [new Agent/UDP]\n",i,k);
            print("$udp%d_%d set class_ 2\n",i,k);
            print("$ns attach-agent $n%d $udp%d_%d\n",i,i,k);
            print("set server%d_%d [new
Application/Traffic/CounterStrikeServer]\n",i,k);
            print("$server%d_%d attach-agent $udp%d_%d\n\n",i,k,i,k);
            k++;
        }
    }

    for(i=n_serv;i<n_serv+n_clientes;i++){
        /* Criando agentes nulos (traffic sink).*/
        print("set null%d [new Agent/Null]\n",i);
        print("$ns attach-agent $n%d $null%d\n\n",i,i);
    }

    /* Conecta-se cada agente UDP de transmissão dos pacotes servidor-
>cliente ao traffic sink do cliente*/
    k=n_serv;
    for(i=0 ; i<n_serv;i++){
        for(j=0;j<v_serv[i];j++){
            print("$ns connect $udp%d_%d $null%d\n",i,k,k);
            k++;
        }
    }

    /* É necessário discriminar um tempo de início de transmissão de dados
e um de término para cada
servidor e cada cliente. */
    for(i=n_serv;i<n_serv+n_clientes;i++)
        print("$ns at 0.1 \"$cliente%d start\"\n",i);
    k=n_serv;
    for(i=0 ; i<n_serv;i++){
        for(j=0;j<v_serv[i];j++){
            print("$ns at 0.1 \"$server%d_%d start\"\n",i,k);
            k++;
        }
    }
    for(i=n_serv;i<n_serv+n_clientes;i++)
        print("$ns at 2.0 \"$cliente%d stop\"\n",i);
    k = n_serv ;
    for(i=0 ; i<n_serv;i++){
        for(j=0;j<v_serv[i];j++){
            print("$ns at 2.0 \"$server%d_%d stop\"\n",i,k);
            k++;
        }
    }

    /* Define o fim da simulação */
    print("$ns at 2.5 \"finish\"\n");
    print("$ns run\n");

    return 0;

```

```

}

/* A função banda devolve um tamanho em Mb para uma banda, esse tamanho é
gerado aleatoriamente respeitando a seguinte proporção:
- 20% serão de 1Mb
- 20% serão de 2Mb
- 20% serão de 34Mb
- 30% serão de 156Mb
- 10% serão de 1Gb
*/
double banda(){

    int x = rand()%10;
    if(x<=1)
        return 1.0;
    if(x<=3)
        return 2.0;
    if(x<=5)
        return 34.0;
    if(x<=8)
        return 156.0;
    return 1000.0;
}

/* função que inicializa a estrutura Union-Find */
void inicializaUF(int pai[MAX_NODES], int sz[MAX_NODES],int n){
    int i;
    for( i=0;i<n;i++){
        pai[i]=i;
        sz[i]=1;
    }
}

/* Função que acha a classe de um elemento no Union-Find*/
int find(int pai[MAX_NODES], int x){
    if(x!=pai[x])
        pai[x] = find(pai,pai[x]);
    return pai[x];
}

/* função que faz a união de duas classe no Union-Find */
void unionUF(int pai[MAX_NODES],int sz[MAX_NODES],int n,int x, int y){
    x = find(pai,x);
    y = find(pai,y);
    if(sz[y]<sz[x]){
        pai[y]=x;
        sz[x] +=sz[y];
        sz[y]=0;
    }
    else{
        pai[x]=y;
        sz[y]+=sz[x];
        sz[x]=0;
    }
}

```

APÊNDICE B – Programa Cliente Counter Strike

```

#include <stdlib.h>
#include <string.h>

#include "../tools/random.h"
#include "../tools/trafgen.h"
#include "../tools/ranvar.h"

class CounterStrikeClient : public TrafficGenerator {
public:
    CounterStrikeClient();
    virtual void timeout();
    virtual double next_interval(int&);
protected:
    virtual void start();
    virtual void stop();
    void init();

    double interval1;
    double interval2;
    int nextInterval;
    double t;
    double interval_;
};

static class CounterStrikeClientClass : public TclClass {
public:
    CounterStrikeClientClass() :
    TclClass("Application/Traffic/CounterStrikeClient") {}
    TclObject* create(int, const char*const*) {
        return (new CounterStrikeClient());
    }
} class_CounterStrikeClient_traffic;

CounterStrikeClient::CounterStrikeClient()
{
    /* 50% dos pacotes que saem dos clientes são mandados a cada 33ms, e
    50% a cada 50ms */
    interval1 = 0.033;
    interval2 = 0.05;
    nextInterval = 1;
}

void CounterStrikeClient::init()
{
    /* Definindo o tipod e pacote como UDP */
    if (agent_)
        agent_>set_pkttype(PT_UDP);
}

void CounterStrikeClient::start()
{
    init();
    running_ = 1;
    timeout();
}

double CounterStrikeClient::next_interval(int& size)

```

```

{
    t += interval_;
    size = size_;
    return(t);
}

void CounterStrikeClient::timeout()
{
    if (! running_)
        return;
    double sample;
    /* Os tamanhos dos pacotes são escolhidos aleatoriamente seguindo a
distribuição normal com parâmetros
    obtidos experimentalmente como explicado na tese. */
    sample = Random::normal (67.35, 8.78);
    size_ = (int)sample;
    send(size_);

    /* Descobrindo o intervalo até o próximo pacote e agendando esse novo
pacote */
    if (nextInterval == 1) {
        timer_.resched(interval1);
        nextInterval = 2;
    }
    else {
        timer_.resched(interval2);
        nextInterval = 1;
    }
}

void CounterStrikeClient::stop () {
    timer_.cancel();
    return;
}

```


APÊNDICE C – Programa servidor Counter Strike

```

#include <stdlib.h>
#include <string.h>
#include <math.h>
#include "../tools/random.h"
#include "../tools/trafgen.h"
#include "../tools/ranvar.h"

class CounterStrikeServer : public TrafficGenerator {
public:
    CounterStrikeServer();
    virtual void timeout();
    virtual double next_interval(int&);
protected:
    virtual void start();
    virtual void stop();
    double mi(double, double);
    double sigma(double, double);
    void init();

    double my_interval;
    double t;
    double interval_;
};

static class CounterStrikeServerClass : public TclClass {
public:
    CounterStrikeServerClass() :
    TclClass("Application/Traffic/CounterStrikeServer") {}
    TclObject* create(int, const char*const*) {
        return (new CounterStrikeServer());
    }
} class_CounterStrikeServer_traffic;

CounterStrikeServer::CounterStrikeServer()
{
    /* O servidor de CounterStrike manda pacotes a cada 60 ms */
    my_interval = 0.06;
}

void CounterStrikeServer::init()
{
    /* Definindo o tipo de pacote como UDP */
    if (agent_)
        agent_>set_pkttype(PT_UDP);
}

void CounterStrikeServer::start()
{
    init();
    running_ = 1;
    timeout();
}

double CounterStrikeServer::next_interval(int& size)
{
    t += interval_;
    size = size_;
}

```

```

        return(t);
    }

void CounterStrikeServer::timeout()
{
    if (! running_)
        return;
    double sample;
    /* Os tamanhos dos pacotes são escolhidos aleatoriamente seguindo a
distribuição lognormal com parâmetros
    obtidos experimentalmente como explicado na tese. */
    sample = Random::lognormal (mi(113.31,34.83),sigma(113.31,34.83));
    size_ = (int)sample;
    send(size_);

    /* Agendando próximo pacote */
    nextPkttime_ = Scheduler::instance().clock() + my_interval;
    timer_.resched(my_interval);
}

void CounterStrikeServer::stop () {
    timer_.cancel();
    return;
}

/* Abaixo funções de parametrização dos dados colhidos para a distribuição
lognormal */
double CounterStrikeServer::mi(double mean, double var){
    return log(mean) - 0.5*log(1+var/(mean*mean));
}

double CounterStrikeServer::sigma(double mean,double var){
    return log(1+var/(mean*mean));
}

```

APÊNDICE D – Programa Analisador

```

#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include <math.h>

#define scan( X... )    fscanf( traceFile, X )
#define MAXP 300000 /* Número máximo de pacotes na simulação */
#define MAXN 10000 /* Número máximo de nós */

int main(){
    char    type; /* Tipo da informação sobre o pacote:*/
                /* +: colocado na fila */
                /* -: retirado da fila */
                /* r: recebido */
    double  time; /* Tempo em que o evento do pacote em questão ocorreu */
    int     sourceNode; /* Nó de origem do pacote (instantâneo) */
    int     destinationNode; /* Nó de destino do pacote (instantâneo)*/
    char    packetName[10]; /* Nome do protocolo utilizado pelo pacote
(UDP no caso do CS) */
    int     packetSize; /* Tamanho do pacote */
    char    flags[10]; /* Algumas flags com informações adicionais sobre o
pacote geradas pelo NS2 */
    int     flowID; /* Classe do pacote 1:Cliente->Servidor  2:Servidor->
Cliente */
    int     sourceAddress; /* Endereço de origem do pacote */
    int     sourcePort; /* Porta associada ao nó de origem para aquele
pacote */
    int     destinationAddress; /*Endereço de destino do pacote */
    int     destinationPort; /* Porta associada ao nó de destino para o
pacote em questão */
    int     sequenceNumber; /* Número da sequência da aplicação geradora
do pacote ao qual ele pertence */
    int     uniquePacketID; /* Identificador do pacote */
    int     nServers, nNodes=0, nClients = 0, nRouters; /* Quantidades de
cada tipo de nó */
    double  packetSumSC=0.0, packetSumCS = 0.0; /* Somas dos tamanhos dos
pacotes de cada tipo */
    double  delaySumSC=0.0, delaySumCS = 0.0; /* Somas dos atrasos dos
pacotes de cada tipo */
    int     hopsSumSC=0, hopsSumCS = 0; /* Somas dos saltos dos pacotes de
cada tipo */
    int     nPacketsSC=0, nPacketsCS=0; /* Números dos pacotes de cada
tipo

                                CS = Cliente->Servidor SC =
Servidor->Cliente */
    int     i, j; /* Variáveis auxiliares */
    char    fileName[25]; /* Cadeia de caracteres para guardar o nome do
arquivo a ser analisado */
    FILE    *traceFile; /* Arquivo a ser analisado */
    FILE    *delaySCFile; /* Arquivo criado com dados para gráfico de
atraso por pacote para os pacotes S->C */
    FILE    *delayCSFile; /* Arquivo criado com dados para gráfico de
atraso por pacote para os pacotes C->S */
    FILE    *sizeCSFile; /* Arquivo criado com dados para gráfico de
tamanho por pacote para os pacotes C->S */
    FILE    *sizeSCFile; /* Arquivo criado com dados para gráfico de
tamanho por pacote para os pacotes S->C */
    FILE    *hopsCSFile; /* Arquivo criado com dados para gráfico de saltos
por pacote para os pacotes C->S */

```

```

FILE      *hopsSCFile; /* Arquivo criado com dados para gráfico de saltos
por pacote para os pacotes S->C */
FILE      *d3CSFile; /* Arquivo criado com dados para gráfico 3D de
atraso por pacote
                                por número de saltos para os pacotes C->S */
FILE      *d3SCFile; /* Arquivo criado com dados para gráfico 3D de
atraso por pacote
                                por número de saltos para os pacotes S->C */
FILE      *clientBarsFile; /* Arquivo criado com dados para gráfico de
barra de atraso por clientes */
double    sendTime[MAXP], delay[MAXP]; /* Vetores para cálculo e
armazenagem do atraso associado a cada pacote */
int maxID; /* Variável destinada a guardar o maior identificador de
pacote dentre todos os pacotes */
int mapa[MAXP]; /* Vetor que associa o identificador de um pacote com
seu nó de origem */
int hops[MAXP]; /* Vetor que conta os saltos de cada pacote */
double varDelaySC, varDelayCS; /* Variáveis utilizadas no cálculo da
variância e consequentemente do
                                desvio padrão do atraso no tráfego
simulado do Counter Strike */
double mediaDelaySC, mediaDelayCS; /* Variáveis utilizadas no cálculo
das médias de atraso */
double sumDelayClientSend[MAXN], sumDelayClientRcvd[MAXN]; /* Vetor que
associa a cada cliente a soma
                                dos
atrasos dos pacotes enviados e recebidos */
int nPacketsClientSend[MAXN], nPacketsClientRcvd[MAXN]; /* Vetores que
associam cada cliente ao número de pacotes
                                enviados ou
recebidos */
printf("Digite o nome do arquivo de trace a ser analisado.\n");
scanf("%s", fileName);
traceFile = fopen(fileName, "r");
printf("Digite o número de servidores.\n");
scanf("%d", &nServers);
nNodes += nServers;
for(i=0; i<nServers; i++){
    printf("Digite o número de clientes para o %do.servidor.\n", i+1);
    scanf("%d", &j);
    nClients += j;
    nNodes += j;
}
printf("Digite o número de roteadores (nem clientes nem servidores) na
rede.\n");
scanf("%d", &nRouters);
nNodes += nRouters;

/* Abrindo os arquivos */
delaySCFile = fopen("delaySC.dat", "w");
delayCSFile = fopen("delayCS.dat", "w");
sizeSCFile = fopen("sizeSC.dat", "w");
sizeCSFile = fopen("sizeCS.dat", "w");
hopsSCFile = fopen("hopsSC.dat", "w");
hopsCSFile = fopen("hopsCS.dat", "w");
d3SCFile = fopen("3dSC.dat", "w");
d3CSFile = fopen("3dCS.dat", "w");
clientBarsFile = fopen("clientBars.dat", "w");

/* Inicializando o vetor */
for(i=0; i<MAXP; i++){

```

```

        delay[i]=0.0;
        hops[i]=0;
    }
    for(i=0;i<MAXN;i++){
        sumDelayClientSend[i]=0.0;
        nPacketsClientSend[i]=0;
        sumDelayClientRcvd[i]=0.0;
        nPacketsClientRcvd[i]=0;
    }

    maxID=0;
    while(scan(" %c %lf %d %d %s %d %s %d %d.%d %d.%d %d %d", &type, &time,
&sourceNode, &destinationNode, packetName,
                                                    &packetSize,
flags, &flowID, &sourceAddress, &sourcePort,
&destinationAddress, &destinationPort, &sequenceNumber,
&uniquePacketID)!=-1){
        /* Para cada entrada sobre um pacote no arquivo de trace analisamos
dados importantes */
        if(type=='+' && sourceNode==sourceAddress){
            /* Se o pacote está saindo do endereço inicial marcamos seu
tempo de envio, o contamos e atualizamos algumas
variáveis de contagem */
            sendTime[uniquePacketID] = time;
            mapa[uniquePacketID] = sourceAddress;
            if(sourceNode<nServers){
                packetSumSC += packetSize;
                nPacketsSC++;
                fprintf(sizeSCFile, "%.4f %d\n",time,packetSize);
            }
            else{
                packetSumCS += packetSize;
                nPacketsCS++;
                fprintf(sizeCSFile, "%.4f %d\n",time,packetSize);
            }
        }
        else if (type=='r' && destinationNode==destinationAddress){
            /* se o pacote está sendo recebido em seu destino final,
calculamos o atraso
e atualizamos algumas contagens */
            delay[uniquePacketID] = time - sendTime[uniquePacketID];
            if(flowID==1){
                sumDelayClientSend[sourceAddress] += delay[uniquePacketID];
                nPacketsClientSend[sourceAddress]++;
            }
            else{
                sumDelayClientRcvd[destinationAddress] +=
delay[uniquePacketID];
                nPacketsClientRcvd[destinationAddress]++;
            }
        }
        }

        if(type=='r')
            hops[uniquePacketID]++;

        if(uniquePacketID>maxID)
            maxID = uniquePacketID;
    }

```

```

        for(i=nServers;i<nServers+nClients;i++){
            fprintf(clientBarsFile, "%d %.5f %.5f\n", i-nServers+1,
1000.0*sumDelayClientSend[i]/(1.0*nPacketsClientSend[i]),
1000.0*sumDelayClientRcvd[i]/(1.0*nPacketsClientRcvd[i]));
        }

        printf("Número total de pacotes:%d\n\n", maxID+1);
        for(i=0; i<=maxID; i++){
            if(mapa[i]<nServers){
                fprintf(delaySCFile, "%d %.4f\n", i, delay[i]*1000.0);
                delaySumSC += delay[i];
                fprintf(hopsSCFile, "%d %d\n", i, hops[i]);
                hopsSumSC+=hops[i];
                fprintf(d3SCFile, "%d %.4f %d\n", i, delay[i]*1000.0, hops[i]);
            }
            else{
                fprintf(delayCSFile, "%d %.4f\n", i, delay[i]*1000.0);
                delaySumCS += delay[i];
                fprintf(hopsCSFile, "%d %d\n", i, hops[i]);
                hopsSumCS+=hops[i];
                fprintf(d3CSFile, "%d %.4f %d\n", i, delay[i]*1000.0, hops[i]);
            }
        }

        mediaDelaySC = 1000*(delaySumSC/(1.0*nPacketsSC));
        mediaDelayCS = 1000*(delaySumCS/(1.0*nPacketsCS));
        varDelaySC=varDelayCS=0.0;
        for(i=0; i<=maxID; i++){
            if(mapa[i]<nServers){
                varDelaySC += (delay[i]*1000 - mediaDelaySC)*(delay[i]*1000 -
mediaDelaySC);
            }
            else{
                varDelayCS += (delay[i]*1000 - mediaDelayCS)*(delay[i]*1000 -
mediaDelayCS);
            }
        }

        printf("Média do tamanho dos pacotes S->C: %.5f\n",
packetSumSC/(1.0*nPacketsSC));
        printf("Média do tamanho dos pacotes C->S: %.5f\n\n",
packetSumCS/(1.0*nPacketsCS));
        printf("Média do delay dos pacotes S->C: %.5f\n", mediaDelaySC);
        printf("Desvio padrão do delay dos pacotes S->C:
%.5f\n", sqrt(varDelaySC/(1.0*nPacketsSC)) );
        printf("Média do delay dos pacotes C->S: %.5f\n", mediaDelayCS);
        printf("Desvio padrão do delay dos pacotes C->S: %.5f\n\n",
sqrt(varDelayCS/(1.0*nPacketsCS)));
        printf("Média de saltos dos pacotes S->C:
%.5f\n", (1.0*hopsSumSC)/(1.0*nPacketsSC));
        printf("Média de saltos dos pacotes C->S: %.5f\n",
(1.0*hopsSumCS)/(1.0*nPacketsCS));
        fclose(delaySCFile);
        fclose(delayCSFile);
        fclose(sizeSCFile);
        fclose(sizeCSFile);
        fclose(d3SCFile);
        fclose(d3CSFile);
        return 0;
    }

```

APÊNDICE E – Programa BGP

TCL BGP

```
#Criando o simulador.
set ns [new Simulator]

#Definindo diferentes cores para os fluxos de dados.
$ns color 1 Blue
$ns color 2 Red

#Abrindo o arquivo de trace do NS.
set f [open out.tr w]
$ns trace-all $f

#Abrindo o arquivo de trace para o NAM.
set nf [open out.nam w]
$ns namtrace-all $nf

#Definindo uma rotina de finalizacao.
proc finish {} {
    global ns nf f
    $ns flush-trace
    #Fechando o arquivo de trace do NS.
    close $f
    #Fechando o arquivo de trace para o NAM.
    close $nf
    #Executando o NAM no arquivo de trace.
    exec nam out.nam &
    exit 0
}

$ns node-config -BGP ON
set n0 [$ns node 0:10.0.0.1]
```

```

set n1 [$ns node 0:10.0.1.1]
set n2 [$ns node 0:10.0.2.1]
set n3 [$ns node 0:10.0.3.1]
set n4 [$ns node 0:10.0.4.1]
set n5 [$ns node 0:10.0.5.1]
set n6 [$ns node 0:10.0.6.1]
set n7 [$ns node 0:10.0.7.1]
set n8 [$ns node 1:10.1.8.1]
set n9 [$ns node 2:10.2.9.1]
set n10 [$ns node 1:10.1.10.1]
$ns node-config -BGP OFF

$ns duplex-link $n0 $n1 1Mb 100ms DropTail
$ns duplex-link $n0 $n5 1Mb 100ms DropTail
$ns duplex-link $n1 $n5 1Mb 100ms DropTail
$ns duplex-link $n0 $n2 1Mb 100ms DropTail
$ns duplex-link $n0 $n3 1Mb 100ms DropTail
$ns duplex-link $n0 $n4 1Mb 100ms DropTail
$ns duplex-link $n1 $n2 1Mb 100ms DropTail
$ns duplex-link $n1 $n3 1Mb 100ms DropTail
$ns duplex-link $n1 $n4 1Mb 100ms DropTail
$ns duplex-link $n5 $n6 1Mb 100ms DropTail
$ns duplex-link $n5 $n7 1Mb 100ms DropTail
$ns duplex-link $n8 $n10 1Mb 100ms DropTail
$ns duplex-link $n2 $n8 1Mb 100ms DropTail
$ns duplex-link $n7 $n9 1Mb 100ms DropTail

set bgp_agent0 [$n0 get-bgp-agent]
$bgp_agent0 bgp-id 10.0.0.1
$bgp_agent0 cluster-id 1000
$bgp_agent0 neighbor 10.0.2.1 route-reflector-client
$bgp_agent0 neighbor 10.0.3.1 route-reflector-client
$bgp_agent0 neighbor 10.0.4.1 route-reflector-client

```



```
$bgp_agent0 neighbor 10.0.1.1 remote-as 0
$bgp_agent0 neighbor 10.0.5.1 remote-as 0
```

```
set bgp_agent1 [$n1 get-bgp-agent]
$bgp_agent1 bgp-id 10.0.1.1
$bgp_agent1 cluster-id 1000
$bgp_agent1 neighbor 10.0.2.1 route-reflector-client
$bgp_agent1 neighbor 10.0.3.1 route-reflector-client
$bgp_agent1 neighbor 10.0.4.1 route-reflector-client
$bgp_agent1 neighbor 10.0.0.1 remote-as 0
$bgp_agent1 neighbor 10.0.5.1 remote-as 0
```

```
set bgp_agent5 [$n5 get-bgp-agent]
$bgp_agent5 bgp-id 10.0.5.1
$bgp_agent1 cluster-id 2000
$bgp_agent5 neighbor 10.0.6.1 route-reflector-client
$bgp_agent5 neighbor 10.0.7.1 route-reflector-client
$bgp_agent5 neighbor 10.0.1.1 remote-as 0
$bgp_agent5 neighbor 10.0.0.1 remote-as 0
```

```
set bgp_agent2 [$n2 get-bgp-agent]
$bgp_agent2 bgp-id 10.0.2.1
$bgp_agent2 neighbor 10.0.0.1 remote-as 0
$bgp_agent2 neighbor 10.0.1.1 remote-as 0
$bgp_agent2 neighbor 10.1.8.1 remote-as 1
```

```
set bgp_agent3 [$n3 get-bgp-agent]
$bgp_agent3 bgp-id 10.0.3.1
$bgp_agent3 neighbor 10.0.0.1 remote-as 0
$bgp_agent3 neighbor 10.0.1.1 remote-as 0
```

```
set bgp_agent4 [$n4 get-bgp-agent]
$bgp_agent4 bgp-id 10.0.4.1
```

```

$bgp_agent4 neighbor 10.0.0.1 remote-as 0
$bgp_agent4 neighbor 10.0.1.1 remote-as 0
set bgp_agent6 [$n6 get-bgp-agent]
$bgp_agent6 bgp-id 10.0.6.1
$bgp_agent6 neighbor 10.0.5.1 remote-as 0

set bgp_agent7 [$n7 get-bgp-agent]
$bgp_agent7 bgp-id 10.0.7.1
$bgp_agent7 neighbor 10.0.5.1 remote-as 0
$bgp_agent7 neighbor 10.2.9.1 remote-as 2

set bgp_agent8 [$n8 get-bgp-agent]
$bgp_agent8 bgp-id 10.1.8.1
$bgp_agent8 neighbor 10.0.2.1 remote-as 0
$bgp_agent8 neighbor 10.1.10.1 remote-as 1

set bgp_agent9 [$n9 get-bgp-agent]
$bgp_agent9 bgp-id 10.2.9.1
$bgp_agent9 neighbor 10.0.7.1 remote-as 0

set bgp_agent10 [$n10 get-bgp-agent]
$bgp_agent10 bgp-id 10.1.10.1
$bgp_agent10 neighbor 10.1.8.1 remote-as 1

set udp0 [new Agent/UDP]
$udp0 set class_ 1
$udp0 set dst_addr_ [$n4 strtocaddr 10.1.10.1]
$udp0 set dst_port_ 0

set client1 [new Application/Traffic/CounterStrikeClient]
$client1 attach-agent $udp0
$ns attach-agent $n4 $udp0

```

```
set udp1 [new Agent/UDP]
$udp1 set class_ 2
$udp1 set dst_addr_ [$n3 strtocaddr 10.0.5.1]
$udp1 set dst_port_ 0

set server1 [new Application/Traffic/CounterStrikeServer]
$server1 attach-agent $udp1
$ns attach-agent $n3 $udp1

$ns at 0.23 "$client1 start"
$ns at 0.23 "$server1 start"
$ns at 0.25 "$bgp_agent8 network 10.1.10.0/24"
$ns at 0.35 "$bgp_agent9 network 10.2.9.0/24"

$ns at 20 "$client1 stop"
$ns at 20 "$server1 stop"

$ns at 39.0 "$bgp_agent0 show-routes"
$ns at 39.0 "$bgp_agent1 show-routes"
$ns at 39.0 "$bgp_agent2 show-routes"
$ns at 39.0 "$bgp_agent3 show-routes"
$ns at 39.0 "$bgp_agent4 show-routes"
$ns at 39.0 "$bgp_agent5 show-routes"
$ns at 39.0 "$bgp_agent6 show-routes"
$ns at 39.0 "$bgp_agent7 show-routes"
$ns at 39.0 "$bgp_agent8 show-routes"
$ns at 39.0 "$bgp_agent9 show-routes"
$ns at 39.0 "$bgp_agent10 show-routes"

$ns at 40.0 "finish"

$ns run
```

APÊNDICE F – Caracterização do tráfego

Summarized Report

Start time: 01/11/2009 - 01:00

Elapsed Time: 6 h

Object: Protocol UDP

Way: Source and destination

Sorted by: Packets

IP address to group translation

IP address to subnet translation

Flows to application translation

Source IP	Destination IP	Source port	Destination port	Bytes	Packets	Flows	Taxa (pps)
201.81.82.252	189.19.107.73	27015	27005	6,801,748	69,627	8	3.223
189.19.107.73	201.81.82.252	27005	27015	5,065,786	67,150	8	3.109
189.78.196.134	201.81.82.252	27005	27015	4,720,132	63,970	8	2.962
201.81.82.252	189.78.196.134	27015	27005	4,351,232	42,960	8	1.989
201.14.107.230	201.81.82.252	61826	27015	987,270	12,566	3	0.582
201.81.82.252	201.14.107.230	27015	61826	1,148,219	8,835	2	0.409
201.81.82.252	201.53.100.132	27015	27005	421,693	5,512	1	0.255
201.81.82.252	189.69.16.100	27015	27005	441,438	3,665	1	0.170
189.69.16.100	201.81.82.252	27005	27015	366,104	3,487	2	0.161
201.53.100.132	201.81.82.252	27005	27015	229,495	2,930	1	0.136
201.81.82.252	208.111.171.83	1100	27017	348,560	2,471	69	0.114
201.81.82.252	69.28.145.170	26901	27017	296,160	2,313	69	0.107
208.111.171.83	201.81.82.252	27017	1100	94,448	990	732	0.046
69.28.145.170	201.81.82.252	27017	26901	50,784	780	756	0.036
201.81.82.252	189.83.47.121	27015	60051	55,282	446	1	0.021
189.83.47.121	201.81.82.252	60051	27015	28,887	412	1	0.019
201.81.82.252	201.81.95.255	137	137	27,144	348	29	0.016
201.81.82.252	84.254.74.131	27015	27005	22,269	305	302	0.014
84.254.74.131	201.81.82.252	27005	27015	14,814	305	302	0.014
201.81.82.252	72.165.61.189	27015	27011	19,075	158	79	0.007
201.81.82.252	68.142.72.250	27015	27011	19,075	158	79	0.007
201.81.82.252	201.81.95.255	138	138	23,986	115	56	0.005
201.81.82.252	81.169.175.185	27015	36592	32,154	96	24	0.004
81.169.175.185	201.81.82.252	36592	27015	4,584	96	24	0.004
201.81.82.252	201.11.232.233	27015	52210	31,019	90	1	0.004
72.165.61.189	201.81.82.252	27011	27015	3,634	79	79	0.004
68.142.72.250	201.81.82.252	27011	27015	3,634	79	79	0.004

201.81.82.252	81.169.175.185	27015	2260	21,633	64	16	0.003
81.169.175.185	201.81.82.252	2260	27015	3,056	64	16	0.003
201.22.215.138	201.81.82.252	60068	27015	3,127	59	2	0.003
201.81.82.252	201.22.215.138	27015	60068	6,844	59	2	0.003
201.81.82.252	81.169.175.185	27015	2258	18,662	56	14	0.003
81.169.175.185	201.81.82.252	2258	27015	2,674	56	14	0.003
189.102.48.93	201.81.82.252	2050	27015	2,438	46	17	0.002
201.81.82.252	189.102.48.93	27015	2050	5,336	46	17	0.002
201.81.82.252	201.12.141.4	27015	1052	4,060	35	12	0.002
201.12.141.4	201.81.82.252	1052	27015	1,855	35	12	0.002
200.203.36.111	201.81.82.252	1101	27015	1,643	31	31	0.001
201.81.82.252	200.203.36.111	27015	1101	3,596	31	31	0.001
201.81.82.252	201.52.243.254	27015	1266	3,480	30	13	0.001
201.52.243.254	201.81.82.252	1266	27015	1,590	30	13	0.001
201.95.88.177	201.81.82.252	1575	27015	1,537	29	8	0.001
201.81.82.252	201.95.88.177	27015	1575	3,364	29	8	0.001
201.253.243.103	201.81.82.252	57415	27015	1,431	27	4	0.001
201.81.82.252	201.210.237.99	27015	1137	3,132	27	3	0.001
201.210.237.99	201.81.82.252	1137	27015	1,431	27	3	0.001
201.81.82.252	201.253.243.103	27015	57415	3,132	27	4	0.001
200.203.36.111	201.81.82.252	1648	27015	1,325	25	16	0.001
76.71.11.12	201.81.82.252	3682	27015	1,325	25	4	0.001
201.81.82.252	76.71.11.12	27015	3682	2,900	25	4	0.001

Summarized Report

Start time: 01/11/2009 - 07:00

Elapsed Time: 6 h

Object: Protocol UDP

Way: Source and destination

Sorted by: Packets

IP address to group translation

IP address to subnet translation

Flows to application translation

Source IP	Destination IP	Source port	Destination port	Bytes	Packets	Flows	Taxa (pps)
201.81.82.252	189.62.107.60	27015	27005	459,041	5,216	1	0.241
189.62.107.60	201.81.82.252	27005	27015	461,767	4,444	2	0.206
201.81.82.252	201.15.253.55	27015	50808	288,109	3,736	1	0.173
201.81.82.252	189.10.158.222	27015	50473	335,873	3,531	1	0.163
189.10.158.222	201.81.82.252	50473	27015	235,473	3,220	2	0.149
201.81.82.252	208.111.171.83	1100	27017	343,248	2,486	69	0.115
201.15.253.55	201.81.82.252	50808	27015	263,251	2,444	2	0.113
189.18.64.120	201.81.82.252	59449	27015	172,000	2,311	1	0.107
201.81.82.252	69.28.145.170	26901	27017	295,040	2,307	69	0.107
201.81.82.252	189.18.64.120	27015	59449	165,461	1,662	1	0.077
201.81.82.252	201.40.48.170	27015	27005	104,969	1,544	1	0.071
189.60.17.39	201.81.82.252	64015	27015	181,005	1,377	1	0.064
201.81.82.252	189.60.17.39	27015	64015	96,147	1,154	1	0.053
208.111.171.83	201.81.82.252	27017	1100	97,936	998	727	0.046
194.236.219.183	201.81.82.252	27005	27015	67,741	993	2	0.046
201.81.82.252	91.16.72.92	27015	27005	128,099	992	1	0.046
201.81.82.252	194.236.219.183	27015	27005	90,247	922	1	0.043
69.28.145.170	201.81.82.252	27017	26901	51,296	782	753	0.036
201.81.82.252	201.68.52.229	27015	27005	66,272	680	1	0.031
201.68.52.229	201.81.82.252	27005	27015	41,114	622	1	0.029
201.81.82.252	200.218.169.141	27015	27005	86,256	495	1	0.023
91.16.72.92	201.81.82.252	27005	27015	40,412	473	1	0.022
201.81.82.252	200.206.178.239	27015	60865	51,182	375	1	0.017
201.40.48.170	201.81.82.252	27005	27015	27,424	362	2	0.017
201.81.82.252	201.81.95.255	137	137	28,080	360	30	0.017
201.81.82.252	84.254.74.131	27015	27005	21,144	296	283	0.014
84.254.74.131	201.81.82.252	27005	27015	14,416	296	283	0.014
200.218.169.141	201.81.82.252	27005	27015	24,577	279	2	0.013

200.206.178.239	201.81.82.252	60865	27015	43,576	273	1	0.013
201.81.82.252	72.165.61.189	27015	27011	19,319	160	80	0.007
201.81.82.252	68.142.72.250	27015	27011	19,319	160	80	0.007
201.40.48.170	201.81.82.252	49445	27015	11,144	146	1	0.007
201.81.82.252	201.81.95.255	138	138	24,592	118	55	0.005
201.81.82.252	201.34.193.38	27015	49978	5,920	110	5	0.005
201.34.193.38	201.81.82.252	49978	27015	5,060	110	5	0.005
201.81.82.252	201.34.193.38	27015	49977	12,644	109	5	0.005
201.34.193.38	201.81.82.252	49977	27015	5,777	109	5	0.005
201.81.82.252	81.169.175.185	27015	36592	35,991	108	27	0.005
81.169.175.185	201.81.82.252	36592	27015	5,157	108	27	0.005
201.81.82.252	81.169.175.185	27015	2261	29,512	90	24	0.004
81.169.175.185	201.81.82.252	2261	27015	4,294	90	24	0.004
201.81.82.252	189.83.47.121	27015	60855	17,640	87	1	0.004
72.165.61.189	201.81.82.252	27011	27015	3,680	80	80	0.004
68.142.72.250	201.81.82.252	27011	27015	3,680	80	80	0.004
10.14.0.1	255.255.255.255	67	68	18,494	55	25	0.003
10.221.192.1	255.255.255.255	67	68	22,825	55	13	0.003
200.194.163.70	201.81.82.252	1311	27015	2,438	46	9	0.002
201.81.82.252	200.194.163.70	27015	1311	5,336	46	9	0.002
201.81.96.1	255.255.255.255	67	68	11,808	36	27	0.002
10.238.192.1	255.255.255.255	67	68	13,310	32	7	0.001

Summarized Report

Start time: 01/11/2009 - 14:00

Elapsed Time: 6 h

Object: Protocol UDP

Way: Source and destination

Sorted by: Packets

IP address to group translation

IP address to subnet translation

Flows to application translation

Source IP	Destination IP	Source port	Destination port	Bytes	Packets	Flows	Taxa (pps)
201.83.179.240	201.81.82.252	63565	27015	680,707	9,073	1	0.420
200.102.96.22	201.81.82.252	50060	27015	224,037	3,185	1	0.147
201.81.82.252	201.95.82.251	27015	27005	300,845	2,868	1	0.133
201.81.82.252	201.83.179.240	27015	63565	304,211	2,821	1	0.131
201.81.82.252	208.111.171.83	1100	27017	346,016	2,433	68	0.113
201.95.82.251	201.81.82.252	27005	27015	269,492	2,326	1	0.108
201.81.82.252	200.102.96.22	27015	50060	209,303	2,245	1	0.104
201.81.82.252	69.28.145.170	26901	27017	285,824	2,234	67	0.103
189.83.47.121	201.81.82.252	61477	27015	95,437	1,486	1	0.069
200.161.125.105	201.81.82.252	62180	27015	104,296	1,465	1	0.068
201.81.82.252	189.83.47.121	27015	61477	113,237	1,417	1	0.066
201.81.82.252	200.161.125.105	27015	62180	99,373	1,038	1	0.048
201.80.104.98	201.81.82.252	27005	27015	67,506	1,032	2	0.048
208.111.171.83	201.81.82.252	27017	1100	100,864	996	709	0.046
201.81.82.252	201.80.104.98	27015	27005	77,021	847	1	0.039
201.58.14.161	201.81.82.252	27005	27015	143,445	823	1	0.038
69.28.145.170	201.81.82.252	27017	26901	50,112	762	734	0.035
201.81.82.252	201.58.14.161	27015	27005	70,747	627	1	0.029
201.81.82.252	201.143.156.85	27015	27005	57,963	614	1	0.028
201.81.82.252	200.172.95.63	27015	61315	63,716	594	1	0.028
201.81.82.252	189.54.53.47	27015	43622	96,689	591	1	0.027
201.143.156.85	201.81.82.252	27005	27015	35,367	532	1	0.025
200.172.95.63	201.81.82.252	61315	27015	121,031	454	1	0.021
189.54.53.47	201.81.82.252	43622	27015	137,689	434	1	0.020
201.81.82.252	189.106.185.40	27015	27005	48,773	429	1	0.020
201.81.82.252	201.81.95.255	137	137	27,144	348	29	0.016
189.106.185.40	201.81.82.252	27005	27015	76,625	304	1	0.014

201.81.82.252	84.254.74.131	27015	27005	19,010	265	258	0.012
84.254.74.131	201.81.82.252	27005	27015	12,918	265	258	0.012
201.81.82.252	189.32.55.27	27015	27005	38,720	257	1	0.012
189.32.55.27	201.81.82.252	27005	27015	65,042	203	1	0.009
201.81.82.252	68.142.72.250	27015	27011	21,254	176	88	0.008
201.81.82.252	72.165.61.189	27015	27011	21,058	175	88	0.008
201.81.82.252	201.81.95.255	138	138	23,986	115	53	0.005
201.81.82.252	81.169.175.185	27015	36592	34,658	104	26	0.005
81.169.175.185	201.81.82.252	36592	27015	4,966	104	26	0.005
201.81.82.252	201.68.36.148	27015	27005	24,656	93	1	0.004
68.142.72.250	201.81.82.252	27011	27015	4,048	88	88	0.004
72.165.61.189	201.81.82.252	27011	27015	4,002	87	87	0.004
201.81.82.252	189.18.176.36	27015	64915	7,888	68	1	0.003
189.18.176.36	201.81.82.252	64915	27015	3,604	68	1	0.003
201.78.69.44	201.81.82.252	1062	27015	2,862	54	5	0.003
201.81.82.252	201.78.69.44	27015	1062	6,264	54	5	0.003
201.83.160.74	201.81.82.252	4160	27015	2,809	53	44	0.002
201.81.82.252	201.83.160.74	27015	4160	6,148	53	44	0.002
201.81.82.252	81.169.175.185	27015	2269	16,033	48	12	0.002
81.169.175.185	201.81.82.252	2269	27015	2,292	48	12	0.002
201.81.112.1	255.255.255.255	67	68	15,088	46	29	0.002
201.81.82.252	201.48.219.39	27015	62485	5,336	46	21	0.002
201.48.219.39	201.81.82.252	62485	27015	2,438	46	21	0.002

Summarized Report

Start time: 02/11/2009 - 10:20

Elapsed Time: 6 h

Object: Device 127.0.0.1

Way: Source and destination

Sorted by: Bytes

IP address to group translation

IP address to subnet translation

Flows to application translation

Source IP	Destination IP	Source port	Destination port	Bytes	Packets	Flows	Taxa (Kbps)
192.168.0.23	192.168.0.25	27015	27005	98,102,147	933,114	34	36.334
192.168.0.23	192.168.0.33	27015	27005	92,864,421	819,908	31	34.394
192.168.0.23	192.168.0.2	27015	27005	84,780,369	774,007	30	31.400
192.168.0.23	192.168.0.24	27015	27005	77,789,595	676,283	26	28.811
192.168.0.23	192.168.0.26	27015	27005	74,917,718	634,878	26	27.747
192.168.0.23	192.168.0.29	27015	27005	72,804,887	635,924	30	26.965
192.168.0.42	192.168.0.17	27016	27005	71,806,464	638,646	24	26.595
192.168.0.23	192.168.0.30	27015	27005	71,637,468	598,107	29	26.532
192.168.0.23	192.168.0.32	27015	27005	71,186,489	612,575	27	26.365
192.168.0.23	192.168.0.241	27015	27005	69,849,120	601,950	28	25.870
192.168.0.42	192.168.0.20	27016	27005	69,730,719	628,424	26	25.826
192.168.0.42	192.168.0.16	27016	27005	68,810,569	621,967	23	25.485
192.168.0.42	192.168.0.19	27016	27005	67,300,322	569,360	23	24.926
192.168.0.42	192.168.0.18	27016	27005	67,167,844	575,980	22	24.877
192.168.0.42	192.168.0.2	27016	27005	63,291,820	542,140	20	23.441
192.168.0.33	192.168.0.23	27005	27015	60,312,224	827,699	31	22.338
192.168.0.25	192.168.0.23	27005	27015	60,276,932	822,515	34	22.325
192.168.0.42	192.168.0.33	27016	27005	59,129,605	510,539	19	21.900
192.168.0.42	192.168.0.25	27016	27005	58,505,472	487,393	18	21.669
192.168.0.42	192.168.0.26	27016	27005	57,245,085	490,014	18	21.202
192.168.0.42	192.168.0.24	27016	27005	53,748,792	457,115	18	19.907
192.168.0.24	192.168.0.23	27005	27015	50,286,529	684,691	26	18.625
192.168.0.23	192.168.0.111	27015	27005	49,725,368	405,627	29	18.417
192.168.0.2	192.168.0.23	27005	27015	49,540,986	685,992	30	18.349
192.168.0.23	192.168.0.12	27015	27005	49,197,348	401,918	16	18.221
192.168.0.42	192.168.0.111	27016	27005	47,739,827	383,960	19	17.681
192.168.0.23	192.168.0.14	27015	27005	47,526,416	389,345	15	17.602

192.168.0.23	192.168.0.13	27015	27005	47,296,158	398,733	17	17.517
192.168.0.26	192.168.0.23	27005	27015	47,256,221	637,167	27	17.502
192.168.0.23	192.168.0.11	27015	27005	47,108,849	393,199	15	17.448
192.168.0.42	192.168.0.32	27016	27005	47,046,695	385,884	18	17.425
192.168.0.23	192.168.0.3	27015	27005	46,516,082	403,008	14	17.228
192.168.0.17	192.168.0.42	27005	27016	46,407,009	629,962	24	17.188
192.168.0.23	192.168.0.15	27015	27005	46,336,587	393,112	16	17.162
192.168.0.23	192.168.0.28	27015	27005	45,646,581	407,410	14	16.906
192.168.0.23	192.168.0.4	27015	27005	45,398,934	393,717	15	16.814
192.168.0.20	192.168.0.42	27005	27016	45,363,905	637,966	26	16.801
192.168.0.241	192.168.0.23	27005	27015	45,310,111	611,368	28	16.782
192.168.0.29	192.168.0.23	27005	27015	44,958,621	615,192	32	16.651
192.168.0.30	192.168.0.23	27005	27015	44,467,339	601,702	29	16.469
192.168.0.16	192.168.0.42	27005	27016	43,829,762	608,029	23	16.233
192.168.0.32	192.168.0.23	27005	27015	43,185,821	580,959	27	15.995
192.168.0.18	192.168.0.42	27005	27016	43,051,606	589,546	22	15.945
192.168.0.42	192.168.0.241	27016	27005	42,792,582	351,735	18	15.849
192.168.0.19	192.168.0.42	27005	27016	42,621,618	579,280	24	15.786
192.168.0.23	192.168.0.5	27015	27005	41,984,071	370,069	14	15.550
192.168.0.42	192.168.0.29	27016	27005	41,658,865	349,496	19	15.429
192.168.0.42	192.168.0.30	27016	27005	39,941,384	333,926	18	14.793
192.168.0.2	192.168.0.42	27005	27016	39,462,473	548,682	21	14.616
192.168.0.33	192.168.0.42	27005	27016	38,948,073	522,896	19	14.425

Summarized Report

Start time: 02/11/2009 - 01:20

Elapsed Time: 6 h

Object: Device 127.0.0.1

Way: Source

Sorted by: Bytes

- ☐ IP address to group translation
- ☐ IP address to subnet translation
- ☐ Flows to application translation

Source IP	Destination IP	Source port	Destination port	Bytes	Packets	Flows	Taxa (Kbps)
192.168.0.42	192.168.0.17	27015	27005	142,127,477	1,386,713	57	52.640
192.168.0.42	192.168.0.18	27015	27005	137,170,320	1,284,300	51	50.804
192.168.0.23	192.168.0.32	27016	27005	134,824,479	1,301,972	56	49.935
192.168.0.42	192.168.0.19	27015	27005	131,429,220	1,226,001	51	48.677
192.168.0.42	192.168.0.20	27015	27005	130,535,670	1,226,414	49	48.347
192.168.0.42	192.168.0.16	27015	27005	119,575,878	1,091,313	45	44.287
192.168.0.23	192.168.0.29	27016	27005	108,403,023	1,046,379	43	40.149
192.168.0.23	192.168.0.241	27016	27005	105,703,653	1,034,639	43	39.150
192.168.0.23	192.168.0.38	27015	27005	100,854,689	892,259	36	37.354
192.168.0.23	192.168.0.33	27016	27005	98,900,673	1,001,521	41	36.630
192.168.0.23	192.168.0.37	27015	27005	98,545,442	889,106	37	36.498
192.168.0.23	192.168.0.111	27015	27005	97,158,897	864,465	36	35.985
192.168.0.23	192.168.0.34	27015	27005	95,097,808	828,736	35	35.221
192.168.0.23	192.168.0.35	27015	27005	93,849,301	832,956	34	34.759
192.168.0.23	192.168.0.26	27015	27005	93,643,350	838,533	34	34.683
192.168.0.23	192.168.0.2	27015	27005	93,206,310	853,604	35	34.521
192.168.0.23	192.168.0.24	27015	27005	92,647,029	833,230	34	34.314
192.168.0.23	192.168.0.41	27015	27005	91,945,968	837,591	34	34.054
192.168.0.23	192.168.0.25	27015	27005	91,310,125	830,182	34	33.819
192.168.0.23	192.168.0.30	27016	27005	89,576,282	798,380	35	33.176
192.168.0.23	192.168.0.39	27016	27005	80,896,018	759,565	30	29.961
192.168.0.23	192.168.0.36	27016	27005	80,010,878	759,590	30	29.634
192.168.0.23	192.168.0.40	27016	27005	79,546,970	759,754	30	29.462
192.168.0.17	192.168.0.42	27005	27015	74,904,368	1,252,049	57	27.742
192.168.0.42	192.168.0.3	27015	27005	69,321,885	696,813	28	25.675
192.168.0.18	192.168.0.42	27005	27015	68,526,469	1,154,212	52	25.380
192.168.0.19	192.168.0.42	27005	27015	65,276,965	1,079,759	51	24.177

192.168.0.20	192.168.0.42	27005	27015	63,756,746	1,065,992	50	23.614
192.168.0.32	192.168.0.23	27005	27016	63,621,671	1,067,794	55	23.564
192.168.0.16	192.168.0.42	27005	27015	62,666,836	1,049,594	46	23.210
192.168.0.23	192.168.0.26	27016	27005	60,824,844	522,235	22	22.528
192.168.0.42	192.168.0.28	27015	27005	60,518,289	596,682	24	22.414
192.168.0.23	192.168.0.2	27016	27005	60,293,657	521,721	21	22.331
192.168.0.23	192.168.0.111	27016	27005	59,212,270	511,563	23	21.930
192.168.0.23	192.168.0.24	27016	27005	58,975,725	506,763	21	21.843
192.168.0.23	192.168.0.25	27016	27005	58,764,307	505,375	21	21.765
192.168.0.42	192.168.0.5	27015	27005	55,730,673	465,097	21	20.641
192.168.0.33	192.168.0.23	27005	27016	55,559,468	960,053	41	20.578
192.168.0.241	192.168.0.23	27005	27016	55,184,414	936,583	43	20.439
192.168.0.42	192.168.0.27	27015	27005	54,966,237	505,516	20	20.358
192.168.0.111	192.168.0.23	27005	27015	54,514,141	893,871	36	20.190
192.168.0.29	192.168.0.23	27005	27016	53,993,538	916,317	43	19.998
192.168.0.42	192.168.0.4	27015	27005	51,047,181	449,266	20	18.906
192.168.0.34	192.168.0.23	27005	27015	49,886,546	822,466	35	18.476
192.168.0.24	192.168.0.23	27005	27015	49,414,643	836,204	34	18.302
192.168.0.25	192.168.0.23	27005	27015	48,991,106	816,579	35	18.145
192.168.0.37	192.168.0.23	27005	27015	48,829,853	797,442	36	18.085
192.168.0.38	192.168.0.23	27005	27015	47,266,704	768,480	36	17.506
192.168.0.41	192.168.0.23	27005	27015	46,964,923	770,587	34	17.394
192.168.0.2	192.168.0.23	27005	27015	46,227,949	768,515	35	17.121

APÊNDICE G – Configuração do servidor de Jogo Counter Strike

```
// server name
hostname "Training Server | Dust2 4ever"

// rcon password
rcon_password "cpm"

// server cvars
mp_friendlyfire 0
mp_footsteps 1
mp_autoteambalance 1
mp_autokick 0
mp_flashlight 1
mp_tkpunish 1
mp_forcecamera 1
sv_alltalk 0
sv_pausable 0
sv_cheats 0
sv_consistency 1
sv_allowupload 1
sv_allowdownload 1
sv_maxspeed 320
mp_limitteams 2
mp_hostagepenalty 5
sv_voiceenable 1
mp_allowspectators 1
mp_timelimit 45
mp_chattime 10
sv_timeout 65
```

```
// round specific cvars
mp_freezetime 2
mp_roundtime 3
mp_startmoney 1000
mp_c4timer 45
mp_fraglimit 0
mp_maxrounds 0
mp_winlimit 0
mp_playerid 0
mp_spawnprotectiontime 5

// bandwidth rates/settings
sv_minrate 0
sv_maxrate 0
decalfrequency 10
sv_maxupdaterate 60
sv_minupdaterate 10

// server logging
log on
sv_logbans 0
sv_logecho 1
sv_logfile 1
sv_log_onefile 0

// operation
sv_lan 0
sv_region 2

// execute ban files
exec banned_user.cfg
exec banned_ip.cfg
```

APÊNDICE H – Script Inicial do Python

```
# -*- coding: utf-8 -*-
#!/usr/bin/python

import sys

def filtraRegistros(lixo, saida=rel, protocolo=udp,
destPort=27015, sourcePort=n):
    """Função que filtra os registros do arquivo de log do
    ethereal

    Parametros
    =====
    - arquivo := arquivo que contém o log do ethereal
    - saida := nome do arquivo de saída
    - protocolo := protocolo a ser filtrado
    - sourcePort := porta de entrada
    - destPort := porta de destino

    Retorno
    =====
    - arquivo com os valores principais dos registros
```

No.	Time	Source	Destination
Protocol Info			
1	0.000000	201.81.128.1	Broadcast
ARP	Who has 201.81.121.254? Tell 201.81.112.1		

```

Frame 1 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:30:b8:c8:1a:50, Dst: ff:ff:ff:ff:ff:ff
Address Resolution Protocol (request)
```



```
0000  ff ff ff ff ff ff 00 30 b8 c8 1a 50 08 06 00 01
.....0...P....
```

```
0010  08 00 06 04 00 01 00 30 b8 c8 1a 50 c9 51 70 01
.....0...P.Qp.
```

```
0020  00 00 00 00 00 00 c9 51 79 fe c9 51 e9 69 c9 51
.....Qy..Q.i.Q
```

```
0030  79 fe 09 d9 05 99 f8 61 87 85 00 00
```

```
y.....a....
```

No.	Time	Source	Destination
-----	------	--------	-------------

Protocol Info

71	0.842757	189.18.102.58	201.81.122.52
----	----------	---------------	---------------

UDP Source port: 52017 Destination port: 27015

Frame 71 (67 bytes on wire, 67 bytes captured)

Ethernet II, Src: 00:30:b8:c8:1a:50, Dst: 00:16:d4:d7:73:e2

Internet Protocol, Src Addr: 189.18.102.58 (189.18.102.58),
Dst Addr: 201.81.122.52 (201.81.122.52)

User Datagram Protocol, Src Port: 52017 (52017), Dst Port:
27015 (27015)

Data (25 bytes)

```
0000  00 16 d4 d7 73 e2 00 30 b8 c8 1a 50 08 00 45 00
.....s..0...P..E.
```

```
0010  00 35 56 29 00 00 77 11 86 bc bd 12 66 3a c9 51
.5V)...w.....f:.Q
```

```
0020  7a 34 cb 31 69 87 00 21 a5 66 ff ff ff ff 54 53
z4.1i...!.f....TS
```

```
0030  6f 75 72 63 65 20 45 6e 67 69 6e 65 20 51 75 65    ource
```

Engine Que

```
0040  72 79 00                                           ry.
```

No.	Time	Source	Destination
Protocol Info			
183	3.494814	201.81.122.52	201.9.30.212
UDP	Source port: 27015		Destination port: 1558

Frame 183 (130 bytes on wire, 130 bytes captured)
 Ethernet II, Src: 00:16:d4:d7:73:e2, Dst: 00:30:b8:c8:1a:50
 Internet Protocol, Src Addr: 201.81.122.52 (201.81.122.52),
 Dst Addr: 201.9.30.212 (201.9.30.212)
 User Datagram Protocol, Src Port: 27015 (27015), Dst Port:
 1558 (1558)
 Data (88 bytes)

```

0000  00 30 b8 c8 1a 50 00 16 d4 d7 73 e2 08 00 45 00
      .0...P....s...E.
0010  00 74 69 0c 00 00 80 11 a6 09 c9 51 7a 34 c9 09
      .ti.....Qz4..
0020  1e d4 69 87 06 16 00 60 52 f2 ff ff ff ff 49 07
      ..i....`R.....I.
0030  42 61 72 62 69 65 72 69 73 20 7c 20 44 75 73 74
      Barbieris | Dust
0040  32 20 34 65 76 65 72 00 64 65 5f 64 75 73 74 32    2
      4ever.de_dust2
0050  00 63 73 74 72 69 6b 65 00 43 6f 75 6e 74 65 72
      .cstrike.Counter
0060  2d 53 74 72 69 6b 65 3a 20 53 6f 75 72 63 65 00    -
      Strike: Source.
0070  f0 00 00 14 00 64 77 00 01 31 2e 30 2e 30 2e 33
      .....dw..1.0.0.3
0080  34 00
      4.

```

O padrão do registro no arquivo de log varia de acordo com o protocolo utilizado no envio e recebimento de pacotes e também com o tamanho dos pacotes como podemos observar no exemplo acima. Então para capturar o início do registro é necessário encontrar o cabeçalho do mesmo.

Na linha seguinte temos os valores dos campos descritos pelo cabeçalho e nas linhas subsequentes detalhes da conexão e códigos hexadecimal descrevendo o pacote de dados.

```
"""
import pdb; pdb.set_trace()

FILTROS = 0
if protocolo: FILTROS += 1
if sourcePort: FILTROS += 1
if destPort: FILTROS += 1

INICIO_REGISTRO = "No.      Time      Source
Destination      Protocol Info"

log = open(arquivo, 'r')
linhas = log.readlines()

registros = [INICIO_REGISTRO + '\n',]
i = 1
filtra = 0
for linha in linhas:
    # Aqui capturamos o inicio do registro
    if linha.strip() == INICIO_REGISTRO:
        count = 1
    # As informações estão na segunda linha do registro
```

```

        # quando a variável count vale 2, estamos na segunda
linha
    if count == 2:
        #Se encontramos o padrão que procuramos
        #incrementamos a variavel filtra
        if protocolo:
            if linha.find(protocolo) >= 0:
                filtra += 1
        if sourcePort:
            if linha.find('Source port: %s' % sourcePort)
>= 0:
                filtra += 1
        if destPort:
            if linha.find('Destination port: %s' %
destPort) >= 0:
                filtra += 1

        # Se filtra é nulo, encontramos o registro que
procuramos
        if filtra == FILTROS:
            # Armazenamos este registro na lista de registros
            registros.append(linha)
        # Reinicializamos as variaveis auxiliares
        i += 1
        count += 1
        filtra = 0

    if not saida:
        saida = arquivo + '.out'
        output = open(saida, 'w')
        output.writelines(registros)
        output.close()

```

```
if __name__ == '__main__':

    filename = sys.argv[1]
    protocolo = sys.argv[2]
    sourcePort = sys.argv[3]
    if not filename or not protocolo or not sourcePort or not
destPort:
        sys.exit('Nothing to do\nUsage: relatorio.py filename
protocolo sourcePort')

    filtraRegistros(filename, protocolo=protocolo,
sourcePort=sourcePort)
```

APÊNDICE I – Configuração Experimentos Ponto de Troca de Tráfego

Configuração e comandos executados no IXP em produção

Configuração do Zebra

Current configuration:

```
!  
hostname IXP  
log file bgpd.log  
log stdout  
!  
bgp multiple-instance  
!  
router bgp 65500  
  bgp router-id 200.215.186.2  
  neighbor 200.215.186.3 remote-as 14026  
  neighbor 200.215.186.3 filter-list provedor1 in  
  neighbor 200.215.186.3 filter-list global out  
  neighbor 200.215.186.3 attribute-unchanged as-path next-hop  
  neighbor 200.215.186.4 remote-as 16397  
  neighbor 200.215.186.4 filter-list provedor2 in  
  neighbor 200.215.186.4 filter-list global out  
  neighbor 200.215.186.4 attribute-unchanged as-path next-hop  
  neighbor 200.215.186.5 remote-as 18747  
  neighbor 200.215.186.5 filter-list provedor3 in  
  neighbor 200.215.186.5 filter-list global out  
  neighbor 200.215.186.5 attribute-unchanged as-path next-hop  
  neighbor 200.215.186.6 remote-as 14346  
  neighbor 200.215.186.6 filter-list provedor4 in  
  neighbor 200.215.186.6 filter-list global out  
  neighbor 200.215.186.6 attribute-unchanged as-path next-hop  
  neighbor 200.215.186.7 remote-as 13878  
  neighbor 200.215.186.7 filter-list provedor5 in
```

```

neighbor 200.215.186.7 filter-list globalidc out
neighbor 200.215.186.7 attribute-unchanged as-path next-hop
neighbor 200.215.186.9 remote-as 13591
neighbor 200.215.186.9 filter-list provedor6 in
neighbor 200.215.186.9 attribute-unchanged as-path next-hop
neighbor 200.215.186.10 remote-as 15180

```

```

neighbor 200.215.186.10 filter-list provedor7 in
neighbor 200.215.186.10 attribute-unchanged as-path next-hop
neighbor 200.215.186.11 remote-as 14571
neighbor 200.215.186.11 filter-list provedor8 in
    neighbor 200.215.186.11 filter-list global out
neighbor 200.215.186.11 attribute-unchanged as-path next-hop

```

```

access-list 9 permit 200.189.167.6
access-list 9 permit 200.202.114.16
access-list 9 permit 200.215.179.32
access-list 9 permit 200.215.186.2
access-list 9 permit 200.215.186.25
access-list 9 permit 200.215.186.26
access-list 9 deny any

```

```
!
```

```

ip as-path access-list provedor4 permit ^14346$
    ip as-path access-list provedor2 permit ^16397$
ip as-path access-list provedor7 permit ^15180_
ip as-path access-list global permit ^14346$
ip as-path access-list global permit ^19182$
ip as-path access-list global permit ^16397$
ip as-path access-list global permit ^15180$
ip as-path access-list global permit ^16735$
ip as-path access-list global permit ^13878$
ip as-path access-list global permit ^18747$
ip as-path access-list global permit ^14571$

```

```

ip as-path access-list global permit ^13591$
ip as-path access-list global permit ^5772$
ip as-path access-list global permit ^14026$
ip as-path access-list globalidc permit ^14346$
ip as-path access-list globalidc permit ^19182$
ip as-path access-list globalidc permit ^16397$
ip as-path access-list globalidc permit ^18747$
ip as-path access-list globalidc permit ^14571$
ip as-path access-list globalidc permit ^13591_
ip as-path access-list globalidc permit ^5772$
ip as-path access-list globalidc permit ^14026$
ip as-path access-list provedor5 permit ^13878$
ip as-path access-list provedor3 permit ^18747$

ip as-path access-list provedor8 permit ^14571$
ip as-path access-list provedor6 permit ^13591_
    ip as-path access-list provedor1 permit ^14026$
!
line vty
    access-class 9
!
end

```

Relação de participantes

```

sh ip bgp summary
BGP router identifier 200.215.186.2, local AS number 65500
38 BGP AS-PATH entries
0 BGP community entries

```


Neighbor	V	AS	MsgRcvd	MsgSent	TblVer	InQ	OutQ	Up/Down	
State/PfxRcd									
200.215.186.3	4	14026	0	0	0	0	0	00:01:26 Active	
200.215.186.4	4	16397	50493	50582	0	0	0	05w0d00h	2
200.215.186.5	4	18747	49358	49540	0	0	0	6d00h23m	11
200.215.186.6	4	14346	0	0	0	0	0	00:01:21 Active	
200.215.186.7	4	13878	50504	50631	0	0	0	04w6d23h	51
200.215.186.9	4	13591	50982	50647	0	0	0	05w0d00h	50
200.215.186.10	4	15180	50755	50704	0	0	0	02w6d06h	33
200.215.186.11	4	14571	50431	50613	0	0	0	05w0d00h	0
200.215.186.13	4	19182	0	0	0	0	0	00:01:31 Active	
200.215.186.14	4	5772	50447	50612	0	0	0	05w0d00h	2
200.215.186.15	4	16735	50454	50597	0	0	0	05w0d00h	20

Total number of neighbors 11

Todas as rotas BGP recebidas

sh ip bgp

BGP table version is 0, local router ID is 200.215.186.2

Status codes: s suppressed, d damped, h history, * valid, > best, i - internal

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 63.215.188.0/22	200.215.186.7			0	13878 i
*> 63.250.176.0/23	200.215.186.9			0	13591 3300 i
*> 63.250.180.0/23	200.215.186.9			0	13591 3300 i
*> 129.222.32.0/19	200.215.186.9			0	13591 5772 i
*> 129.222.64.0/19	200.215.186.9			0	13591 5772 i
*> 148.188.174.0/24	200.215.186.9			0	13591 3300 i
*> 148.243.64.0/21	200.215.186.7			0	13878 i
*> 157.253.0.0	200.215.186.7			0	13878 i
*> 172.16.32.0/20	200.215.186.7			0	13878 i

*> 172.16.48.0/20	200.215.186.7		0 13878 i
*> 172.16.64.0/20	200.215.186.7	0	0 13878 i
*> 192.5.5.0	200.215.186.9		0 13591 22548 30122 3557 i
*> 192.175.48.0	200.215.186.9		0 13591 22548 112 i
*> 192.228.80.0	200.215.186.9		0 13591 22548 30122 i
*> 200.3.244.0	200.215.186.7		0 13878 i
*> 200.3.245.0	200.215.186.7		0 13878 i
*> 200.3.246.0	200.215.186.7		0 13878 i
*> 200.3.247.0	200.215.186.7		0 13878 i
*> 200.23.192.0/22	200.215.186.7		0 13878 i
*> 200.23.196.0	200.215.186.7		0 13878 i
*> 200.23.197.0	200.215.186.7		0 13878 i
*> 200.31.64.0	200.215.186.7		0 13878 i
*> 200.31.65.0	200.215.186.7		0 13878 i
*> 200.31.66.0	200.215.186.7		0 13878 i
*> 200.31.67.0	200.215.186.7		0 13878 i
*> 200.31.68.0	200.215.186.7		0 13878 i
*> 200.31.69.0	200.215.186.7		0 13878 i
*> 200.31.70.0	200.215.186.7		0 13878 i
*> 200.31.71.0	200.215.186.7		0 13878 i
*> 200.31.72.0	200.215.186.7		0 13878 i
*> 200.31.73.0	200.215.186.7		0 13878 i
*> 200.31.74.0	200.215.186.7		0 13878 i
*> 200.31.76.0	200.215.186.7		0 13878 i
*> 200.31.77.0	200.215.186.7		0 13878 i
*> 200.31.78.0	200.215.186.7		0 13878 i
*> 200.31.79.0	200.215.186.7		0 13878 i
*> 200.31.80.0	200.215.186.7		0 13878 i
*> 200.31.81.0	200.215.186.7		0 13878 i
*> 200.31.83.0	200.215.186.7		0 13878 i
*> 200.31.84.0	200.215.186.7		0 13878 i
*> 200.31.85.0	200.215.186.7		0 13878 i

*> 200.31.86.0	200.215.186.7		0 13878 i
*> 200.49.77.0	200.215.186.9		0 13591 i
*> 200.57.32.0/20	200.215.186.7		0 13878 i
*> 200.57.36.0/22	200.215.186.7		0 13878 i
*> 200.99.0.0/17	200.215.186.10		0 15180 i
*> 200.99.0.0/18	200.215.186.10		0 15180 i
*> 200.99.64.0/18	200.215.186.10		0 15180 i
*> 200.142.64.0/19	200.215.186.9		0 13591 i
*> 200.142.95.0	200.215.186.9		0 13591 i
*> 200.142.160.0/20	200.215.186.10		0 15180 26104 i
*> 200.142.208.0/20	200.215.186.9		0 13591 16397 26602 i
*> 200.142.224.0/20	200.215.186.10		0 15180 26090 i
*> 200.143.0.0/19	200.215.186.7		0 13878 i
*> 200.143.0.0/22	200.215.186.7	0	0 13878 i
*> 200.146.192.0/18	200.215.186.15		0 16735 i
*> 200.150.0.0/21	200.215.186.10		0 15180 23106 i
*> 200.150.192.0/20	200.215.186.9		0 13591 26107 i
*> 200.152.48.0/20	200.215.186.10		0 15180 26616 i
*> 200.152.48.0/22	200.215.186.10		0 15180 26616 i
*> 200.152.52.0/22	200.215.186.10		0 15180 26616 i
*> 200.152.192.0/19	200.215.186.9		0 13591 i
*> 200.155.0.0/19	200.215.186.4		0 16397 i
*> 200.155.15.0	200.215.186.9		0 13591 16397 i
*> 200.155.16.0/20	200.215.186.9		0 13591 16397 i
*> 200.155.96.0/20	200.215.186.9		0 13591 23002 i
*> 200.155.98.0	200.215.186.9		0 13591 23002 ?
*> 200.155.107.0	200.215.186.9		0 13591 23002 i
*> 200.155.108.0	200.215.186.9		0 13591 23002 i
*> 200.155.128.0/18	200.215.186.5		0 18747 i
*> 200.155.128.0/21	200.215.186.5		0 18747 i
*> 200.155.136.0/22	200.215.186.5		0 18747 i

*> 200.160.0.0/20 200.215.186.9	0 13591 22548 i
*> 200.160.174.0 200.215.186.9	0 13591 22341 i
*> 200.160.224.0/19 200.215.186.9	0 13591 i
*> 200.162.0.0/17 200.215.186.10	0 15180 i
*> 200.162.0.0/18 200.215.186.10	0 15180 i
*> 200.162.64.0/18 200.215.186.10	0 15180 i
*> 200.162.128.0/20 200.215.186.9	0 13591 22177 i
*> 200.162.176.0/20 200.215.186.10	0 15180 22129 i
*> 200.162.176.0 200.215.186.10	0 15180 22129 i
*> 200.162.177.0 200.215.186.10	0 15180 22129 i
*> 200.169.128.0/17 200.215.186.9	0 13591 14650 i
*> 200.170.128.0/18 200.215.186.15	0 16735 i
*> 200.170.128.0/21 200.215.186.15	0 16735 i
*> 200.170.136.0/21 200.215.186.15	0 16735 i
*> 200.170.144.0/21 200.215.186.15	0 16735 i
*> 200.170.152.0/21 200.215.186.15	0 16735 i
*> 200.170.160.0/21 200.215.186.15	0 16735 i
*> 200.170.168.0/21 200.215.186.15	0 16735 i
*> 200.170.176.0/20 200.215.186.15	0 16735 i
*> 200.170.192.0/18 200.215.186.5	0 18747 i
*> 200.170.224.0/20 200.215.186.5	0 18747 i
*> 200.189.96.0/22 200.215.186.9	0 13591 18739 i
*> 200.189.100.0/22 200.215.186.9	0 13591 18739 i
*> 200.189.104.0/22 200.215.186.9	0 13591 18739 i
*> 200.189.108.0/22 200.215.186.9	0 13591 18739 i
*> 200.189.160.0/19 200.215.186.7	0 13878 i
*> 200.189.160.0 200.215.186.7	0 13878 i
*> 200.189.164.0/23 200.215.186.7	0 13878 i
*> 200.192.160.0/20 200.215.186.9	0 13591 19089 13935 i
*> 200.192.216.0/21 200.215.186.9	0 13591 14723 i
*> 200.195.224.0/19 200.215.186.5	0 18747 i

*> 200.195.240.0/22 200.215.186.5	0 18747 i
*> 200.195.246.0/23 200.215.186.5	0 18747 i
*> 200.196.32.0/20 200.215.186.10	0 15180 11431 i
*> 200.196.32.0/22 200.215.186.10	0 15180 11431 i
*> 200.196.36.0/23 200.215.186.10	0 15180 11431 11431 11431 i
*> 200.196.38.0/23 200.215.186.10	0 15180 11431 11431 11431 i
*> 200.196.40.0/23 200.215.186.10	0 15180 11431 11431 11431 i
*> 200.196.42.0/23 200.215.186.10	0 15180 11431 i
*> 200.196.44.0/23 200.215.186.10	0 15180 11431 i
*> 200.196.46.0/23 200.215.186.10	0 15180 11431 11431 11431 i
*> 200.198.64.0/18 200.215.186.10	0 15180 i
*> 200.198.64.0/19 200.215.186.10	0 15180 i
*> 200.198.96.0/19 200.215.186.10	0 15180 i
*> 200.198.176.0/20 200.215.186.4	0 16397 i
*> 200.198.184.0/23 200.215.186.9	0 13591 16397 i
*> 200.201.128.0/19 200.215.186.5	0 18747 i
*> 200.201.144.0/20 200.215.186.5	0 18747 i
*> 200.202.112.0/20 200.215.186.10	0 15180 i
*> 200.202.112.0/21 200.215.186.10	0 15180 i
*> 200.202.120.0/21 200.215.186.10	0 15180 i
*> 200.215.128.0/19 200.215.186.9	0 13591 10688 i
*> 200.215.176.0/20 200.215.186.7	0 13878 i
*> 200.215.176.0/22 200.215.186.7	0 13878 i
*> 200.215.180.0/22 200.215.186.7	0 13878 i
*> 200.215.182.0/23 200.215.186.7	0 13878 i
*> 200.215.182.0 200.215.186.7	0 13878 i
*> 200.215.184.0/22 200.215.186.7	0 13878 i
*> 200.215.187.0 200.215.186.7	0 13878 i
*> 200.215.208.0/20 200.215.186.5	0 18747 i
*> 200.219.0.0/18 200.215.186.9	0 13591 16397 6543 i

```

*> 200.219.64.0/18 200.215.186.9          0 13591 14650 i

*> 200.219.128.0 200.215.186.9          0 13591 7313 7313 7313 7313
7313 7313 7313 7313 7313 7313 7313 i

*> 200.219.224.0/23 200.215.186.10      0 15180 14026 i
* 200.220.0.0/18 200.215.186.9          0 13591 5772 i
*>          200.215.186.14      0      0 5772 i
* 200.220.64.0/18 200.215.186.9          0 13591 5772 i
*>          200.215.186.14      0      0 5772 i
*> 200.220.141.0 200.215.186.10          0 15180 27693 i
*> 200.220.142.0 200.215.186.10          0 15180 27693 i
*> 200.225.64.0/20 200.215.186.9          0 13591 i
*> 200.225.80.0/20 200.215.186.9          0 13591 19089 i
*> 200.225.80.0/22 200.215.186.9          0 13591 19089 i
*> 200.225.84.0/22 200.215.186.9          0 13591 19089 i
*> 200.225.88.0/22 200.215.186.9          0 13591 19089 i
*> 200.225.92.0/22 200.215.186.9          0 13591 19089 i
*> 200.225.160.0/19 200.215.186.9          0 13591 14650 i
*> 200.225.192.0/18 200.215.186.15        0 16735 i
*> 200.225.192.0/21 200.215.186.15        0 16735 i
*> 200.225.200.0/21 200.215.186.15        0 16735 i
*> 200.225.208.0/21 200.215.186.15        0 16735 i
*> 200.225.212.0/22 200.215.186.15        0 16735 i
*> 200.225.216.0/21 200.215.186.15        0 16735 i
*> 200.225.224.0/21 200.215.186.15        0 16735 i
*> 200.225.232.0/21 200.215.186.15        0 16735 i

*> 200.225.240.0/21 200.215.186.15        0 16735 i
*> 200.225.248.0/21 200.215.186.15        0 16735 i
*> 200.229.0.0/20 200.215.186.10          0 15180 16712 16712 16712 i
*> 200.229.16.0/20 200.215.186.9          0 13591 16736 i
*> 200.229.16.0/22 200.215.186.9          0 13591 16736 i
*> 200.229.20.0/22 200.215.186.9          0 13591 16736 i

```

*> 200.229.24.0/22	200.215.186.9	0 13591 16736 i
*> 200.233.128.0/18	200.215.186.15	0 16735 i
*> 207.83.96.0/20	200.215.186.9	0 13591 3300 i
*> 209.58.74.0	200.215.186.7	0 13878 i
*> 209.58.125.0	200.215.186.7	0 13878 i
*> 216.72.229.0	200.215.186.7	0 13878 i

APÊNDICE J – Funcionamento do protocolo BGP, principais comandos de implementação e análise

Cabeçalho do BGP

Segundo Rekhter (1995), “o cabeçalho do BGP é formado com 16 bytes seguidos de 2 bytes destinado ao tamanho e outro byte para identificar seu tipo”. O cabeçalho é usado para fazer processo de estabelecimento de conexão do BGP, ele carrega campos que identificam informações do sistema autônomo.

Todas as mensagens do BGP são compostas de, no mínimo, um cabeçalho e, opcionalmente, uma parte de dados.

O menor pacote de mensagem do BGP não pode ser menor que 19 bytes, ou seja, 16 bytes de cabeçalho, 2 bytes de tamanho e 1 byte de tipo.

No campo do tipo da mensagem existem quatro possibilidades, Open, Update, Notification e Keepalive.

O BGP possui quatro tipos de pacotes de mensagens que utilizam um cabeçalho em comum, como mostrado a seguir:

Mensagem Open

Após estabelecer uma conexão TCP, os vizinhos BGP trocam mensagens open para criar uma conexão BGP. Depois do estabelecimento de uma conexão BGP, os vizinhos trocam outras mensagens BGP e dados com informações de roteamento, conforme Juniper (2003).

Os parâmetros da Mensagem Open descritos por Rekhter (1995) são:

- **Version:** 1 byte que indica a versão do protocolo BGP, que pode ser BGP-3 ou BGP-4. Geralmente, usa-se a versão 4.
- **My Autonomous System:** 2 bytes que indicam o número do AS do roteador que originou a mensagem.
- **Hold Timer:** 2 bytes que indicam o tempo máximo em segundos entre o recebimento sucessivo do *Keepalive* e da mensagem de Update, através de um contador. Quando for recebido um *Keepalive* ou uma mensagem Update, automaticamente o Hold Timer é zerado. Com este mecanismo pode-se perceber se um vizinho está ativo.
- **BGP Identifier:** 4 bytes que indicam o endereço IP, identificando quem enviou. Esta identificação pode ser criada pelo roteador ou pelo endereço do roteador (endereço de *loopback*) mais alto que iniciou a conexão.
- **Optional Parameter Length:** 1 byte que indica a quantidade de bytes do *Optional Parameter field*; caso for 0, indica que não tem esse parâmetro.
- **Optional Parameter:** indica uma lista de parâmetros usados na negociação do BGP. É subdividido em três campos: *parameter type*, *parameter length* e *parameter value*. Estes parâmetros podem ser utilizados para a autenticação do BGP.

Mensagem Update

A mensagem de Update é responsável pelo transporte de roteamento entre os vizinhos BGP. “Cada update contém um anúncio de caminho com seus atributos e destinos.” Juniper (2003). Estas informações estão distribuídas em três partes da mensagem Update. As mensagens com o mesmo atributo devem ser agrupadas em um mesmo Update.

Mensagem Notification

A mensagem de notificação é sempre enviada quando um erro é percebido; após esse evento, a conexão com o vizinho é desfeita. A mensagem Notification consiste de:

- Cabeçalho BGP.
- Código do erro.
- Subcódigo.
- Dados que descrevem o erro.

Tipos	Mensagem
1	Message header error
2	Open message error
3	Update message error
4	Hold timer expired
5	Finite state machine error
6	Cease

Códigos de erro

Mensagens de Keepalive

As mensagens de Keepalive são periodicamente trocadas, para certificar que a conexão continua estabelecida, ou seja, garantir que o vizinho está ativo. O pacote do Keepalive tem 19 *bytes* e não prejudica o uso de banda ou aumento de CPU do equipamento, diferentemente das mensagens de Updates que contêm trocas de informações de redes.

Máquina de estados do BGP

O processo de conexão TCP destinado a trocar mensagens de roteamento está representado pela máquina de estado finita, na qual ocorre o estabelecimento da vizinhança e, posteriormente, a troca de mensagens de roteamento. O conhecimento da máquina de estado é importante para o diagnóstico na rede.

Máquina de estado do BGP, Halabi e Mc Pherson (2000)

A máquina de estado do BGP tem os seguintes estados de negociação:

- **Idle:** é o primeiro estágio da conexão TCP. Esse estágio aguarda um evento para iniciar o processo de conexão.
- **Connect:** aguarda a conexão TCP ser completada. Caso a conexão TCP obtenha sucesso, será enviada uma mensagem OPEN, passando para o estado Open Sent. Caso contrário, irá para o estado Active. No caso do tempo de espera ter sido ultrapassado, o estado volta para Connect. Em qualquer outro evento, retorna-se para Idle.
- **Active:** neste estágio ocorre uma tentativa da conexão TCP. Caso haja sucesso, será enviada uma mensagem de Open, passando para o próximo estado: o OpenSent. Se esta tentativa não for bem-sucedida pelo motivo de expiração do tempo, por exemplo, o estado passa para Connect. Caso não consiga a conexão, serão realizadas outras tentativas, até a expiração do número de tentativas, passando para o estado de Idle.
- **OpenSent:** Aguarda pela mensagem de Open e faz uma checagem de seu conteúdo. Caso seja encontrado algum erro, como número de AS incoerente, não esperado ou a própria versão do BGP, envia-se uma mensagem tipo Notification e volta ao estado de Idle. Quando a mensagem Open é recebida, é

enviado um pacote Keepalive, reinicializando o contador de Keepalives e passando para o estado de OpenConfirm.

- **OpenConfirm:** aguarda a mensagem Keepalive ou Notification. Caso seja recebida a mensagem Keepalive, a conexão será estabelecida, passando para o estado Established. Caso seja recebida a mensagem Notification, o estado passará para Idle.
- **Established** é o último estado de negociação. É nesse estado que pacotes de Update e Keepalive são enviados. Caso seja recebido o pacote de Notification, a conexão será desfeita passando para o estado de Idle.

O estado da máquina de estado do BGP pode ser verificado através dos comandos de análise vistos posteriormente.

Atributos do BGP

O protocolo BGP, ao receber as atualizações de rotas de diferentes AS, repassa somente aquela que for considerada a melhor. O processo de decisão considera em diferentes atributos, como o next-hop, administrative weights, local-preference, route origin, path length, origin code e metric, que serão vistos mais adiante. Estes atributos serão usados para definir a política de roteamento do IXP. A manipulação dos atributos é fundamental para controle do tráfego de entrada e saída de cada participante.

O BGP compara o AS-PATH entre outros atributos, para selecionar o melhor caminho.

Os atributos classificados como well-known devem ser conhecidos pelos vizinhos; os opcionais, não necessariamente. Os atributos well-known são:

- **Mandatory:** são atributos de presença obrigatória em todos os anúncios de prefixos.
 - AS - PATH (TYPE=2)

- ORIGIN (TYPE=1)
- NEXT - HOP (TYPE=3)
- **Discretionary:** são atributos que não necessariamente estão presentes nos anúncios dos prefixos.
 - LOCAL-PREFERENCE (TYPE=5)

Os atributos opcionais são:

- **Transitive:** são propagados para todos os vizinhos. No entanto, o vizinho precisa ter configurado um parâmetro para aceitar esses atributos.
 - COMMUNITY (TYPE=8)
 - AGGREGATOR (TYPE=7)
- **Nontransitive:** caso não reconhecido pelo vizinho, será descartado.
 - MED (TYPE=4)

Tipos de atributos

Os atributos são um conjunto de parâmetros que descrevem as características das rotas. O processo de decisão do BGP faz uso dos atributos para escolher a melhor rota para cada destino. A seguir, apresentam-se os principais atributos:

- **AS-PATH:** quando a rota é propagada pelos AS, a informação do caminho é armazenada. O atributo AS-Path é usado para detecção de *loop*, pois contém a lista de AS percorridos da origem até o destino. É muito importante na definição da política de roteamento.

Exemplo de AS PATH

A Figura 21 indica a rota anunciada pelo AS 100, passando pelo 200 e 400 e chegando ao destino. No AS-PATH a seqüência é registrada como 400, 200, 100.

- **NEXT - HOP:** este parâmetro informa o endereço IP do roteador que anunciou a rota, e não será modificado quando as rotas forem propagadas por uma conexão IBGP.
- **LOCAL-PREFERENCE:** este atributo é conhecido por vizinhos de um AS e usado para determinar o melhor caminho de saída. O caminho que tiver o local-preference maior será escolhido. O valor-padrão para esse atributo é 100.
- **MULTI-EXIT DISCRIMINATOR-MED:** determina o melhor caminho para o tráfego que estará entrando no AS. Para se usar este parâmetro deverá haver um acordo entre os AS(s), pois sua inclusão exige configuração em ambos os lados.
- **WEIGHT:** é um atributo proprietário dos equipamentos Cisco, possibilitando atribuir peso para cada vizinho do roteador não divulgado pelo BGP. Seu valor-padrão é 32768. É para uso interno do próprio roteador.
- **ORIGIN:** este atributo, como visto, é obrigatório, sendo responsável para definir a informação da rota de origem. Este atributo pode ser classificado em:
 - **IGP:** quando a rota for originada dentro do próprio AS pelo comando network do BGP.

- **EGP:** quando a rota for aprendida através do protocolo EGP.
- **Incomplete:** caso a rota tenha sido inserida na TABELA do BGP por redistribuição.
- **COMMUNITY:** é usado para caracterizar um conjunto de rotas de um cliente e controlar o seu anúncio.

O parâmetro Community pode ser usado na estrutura do IXP para distinguir e manipular anúncios de diferentes pontos de trocas, por exemplo, nacionais e internacionais, anunciando nos bloqueios de redes não desejadas

O BGP tem um algoritmo para escolha das rotas. Os atributos do BGP têm diferentes prioridades, de acordo com a Tabela abaixo:

ATRIBUTO	CRITÉRIO
1 – Next-hop	Caso não esteja acessível, não haverá anúncio.
2 – Weight	Maior valor do parâmetro weight.
3 – Local-Preference	Maior valor do local-preference.
4 – (Local)	Em caso de empate: rota originada pelo roteador local.
5 – As-path	Caso não seja interna: rota com o As-path mais curto.
6 – Origin	Melhor precedência (IGP<EGP<INCOMPLETE).
7 – MED	Menor valor do parâmetro MED.
8 – EBGp –IBGP	Caminho externo.
9 – Vizinho mais próximo	Vizinho mais próximo (<i>next-hop</i>).
10 – Rota antiga	Rota mais antiga.
11 – Route ID	Menor valor do router ID.

Prioridade do atributo do BGP

Sincronização do BGP

Para evitar *loops* e “buracos negros”, quando o tráfego é roteado internamente e não atinge o seu objetivo, usa-se a sincronização do BGP.

O BGP precisa estar sincronizado com o protocolo interno (IGP). Isto significa que o BGP espera que a rota seja aprendida pelo IGP antes de divulgá-la para os vizinhos. Pode-se desabilitar a sincronização quando todos os roteadores rodam BGP entre si. A sincronização é habilitada por default, é recomendado mantê-la ativa, existem provedores que colocam rotas estáticas para uma interface nula para garantir o anúncio independente de aprender a rota internamente.

Configurando BGP

Os dois exemplos de configuração estão relacionados a equipamentos Cisco. Posteriormente serão vistos os comandos nos servidores de roteamento Zebra e Gated.

Comandos iniciais

O primeiro passo para ativação do BGP é definir o AS a que o roteador pertence. Pode-se fazer isso com o comando **Router bgp**.

- **Comando: Router bgp** *Autonomous System*

Ex.: Router bgp 100 (este comando inicia o processo BGP relacionado ao AS 100). O número do AS pode ser privado ou público de acordo com a operadora.

O próximo passo é informar quais rotas deverão ser divulgadas pelo BGP através do comando **Network**. No caso de roteadores Cisco, se a sincronização estiver habilitada, só serão divulgadas se estiverem presentes na Tabela de roteamento IP.

- **Comando: Network** *número da rede* **mask** *máscara da rede*.

Ex.: network 10.10.10.0 mask 255.255.255.0 (Anunciando a rede 10.10.10.0 255.255.255.0) O roteador da Cisco usa máscara na forma decimal, não é recomendado anunciar redes menores que uma classe C pois aumentam a Tabela de rotas da Internet.

Como já citado na máquina de estado, os roteadores serão vizinhos quando existir uma conexão TCP entre eles. Estabelecida a conexão, serão trocadas as informações de roteamento.

Para a definição de um vizinho, usa-se o comando **Neighbor**.

- **Comando: Neighbor** *ip address* **remote-as** *as-number*

Ex:::

```
neighbor 10.10.10.2 remote-as 100
```

```
neighbor 172.17.10.1 remote-as 200
```

No exemplo acima define-se o vizinho 10.10.10.2 referente ao AS 100, ou seja, conforme visto é uma seção IBGP (mesmo AS), e o vizinho 172.16.10.1 referente ao AS200, seção EBGP (AS diferente). Para configuração do vizinho é necessário conectividade.

Para identificar o vizinho, utiliza-se o endereço IP da interface ou o endereço da *loopback* (endereço do roteador).

Com os comandos descritos, já se pode configurar o BGP de um roteador Cisco, ou seja, um possível participante do IXP conforme a Tabela 5.

Comandos	Definição
Router bgp 100	Definir processo BGP do AS 100.
Network 10.10.10.0 mask 255.255.255.0	Divulgar a rede 10.10.10.0/24.
Neighbor 10.10.10.2 remote-as 100	Configurar um vizinho do AS.
Neighbor 172.17.10.1 remote-as 200	Configurar um vizinho de outro AS.

Configuração inicial do BGP

Implementando políticas de roteamento no IXP

É fundamental definir a política de roteamento, tanto no IXP quanto nos participantes. Consegue-se manipular os atributos do BGP para controle das rotas enviadas e recebidas, monitorando, desta forma, o tráfego de entrada e saída.

Expressões AS-PATH

O responsável pelo roteamento do AS deve determinar quais blocos serão anunciados e recebidos pelos seus vizinhos, por meio de expressões AS-PATH que serão usadas para criar os filtros de AS nos roteadores Cisco e no servidores de roteamento Zebra.

Foram definidos alguns símbolos para auxiliar a criação dos filtros:

Símbolos	Definição
.	Qualquer caractere unitário.
.^	Especifica a primeira <i>string</i> .
\$	Especifica a última <i>string</i> .
\caractere	Especifica a <i>string</i> em qualquer posição.
[5-9]	Especifica uma faixa de AS.
*	Qualquer especificação, incluindo vazio.
+	Qualquer especificação não vazia.
?	Caractere específico.
-	Separa os extremos de uma faixa de AS.
—	Representa uma vírgula (,) chaves ({ }), parênteses, início da <i>string</i> , fim da <i>string</i> ou espaço em branco.

Símbolos para criação de filtros

Expressões	Definição
.	Representa todo o conjunto de rotas BGP.
^\$	Representa somente as rotas locais do próprio AS.
^100\$	Somente as rotas pertencentes ao AS100.
_100\$	Rotas que foram originadas no AS100.
^100_	Rotas terminadas no AS100.
100	Rotas que atravessaram o AS100.

Exemplos de expressões AS-PATH

Filtros BGP

Os filtros BGP servem para controlar o fluxo de entrada e saída de atualizações de rotas do protocolo, ou seja, eles podem ser aplicados tanto para anúncios enviados (*outbound*) quanto para os recebidos (*inbound*). O uso de filtro é de grande importância no IXP, pois controla as rotas recebidas e enviadas no roteador de cada participante e no servidor de roteamento.

Existem os seguintes métodos de filtragem:

- *AS-Path filtering* (por AS-PATH).
- *Distribute-List filtering* (por lista de acesso).
- *Prefix filtering* (por prefixo).
- *Route Map filtering* (por Route-Map).

Filtro por AS-Path Filtering

Com este filtro podem-se usar as expressões AS-PATH vistas anteriormente, ou seja, podem-se fazer filtros com base nos AS(s). Com o uso de um comando associado a um vizinho, pode-se selecionar rotas de acordo com as condições definidas pelas regras da expressão, permitindo-se ou não sua divulgação. A seguir, uma breve descrição dos comandos.

Comandos de filtro AS-PATH

Comando: `ip as-path access-list` *número da lista de acesso* **deny/permit** *expressão*.

É criada uma lista de acesso relacionada à expressão AS-PATH.

No exemplo a seguir, é mostrada a construção da lista de acesso 1, permitindo que sejam aceitas as redes originadas pelo AS 200.

```
Ip as-path access-list 1 permit ^200$
```

Comando: `Neighbor endereço IP filter-list` *número da lista de acesso* **in/out**

Associar ao vizinho, identificando se o filtro será aplicado na entrada (recebimento) ou saída (divulgação).

No exemplo, a aplicação desta lista na entrada do vizinho 10.10.10.2

```
Neighbor 10.10.10.2 filter-list 1 in
```

A Tabela 8 mostra uma seqüência de comandos executados para configurar um filtro AS-PATH.

Comandos	Definição
<pre>Router bgp 100 Neighbors 172.17.10.1 remote-as 200 neighbors 172.17.10.1 filter-list 10 in Ip as-path access-list 10 permit ^200\$</pre>	<p>Configura o vizinho 172.17.10.1 para receber somente as rotas originadas pelo AS 200.</p>

Configuração de filtro AS-PATH no BGP

Filtro por distribute-list

O filtro por distribute-list precisa da criação de uma lista de acesso estendida, ou seja, precisa ser usado quando se deseja fazer filtros pelo endereço IP. A seguir, a descrição dos comandos.

Comandos de filtro por distribute-list

Comando: **Access-list** *number* **deny/permit** *ip origem ip destino*

É construída uma lista de acesso através dos parâmetros deny (nega) e permit (permite) do endereço origem para destino.

No exemplo, a construção da lista de acesso 199 negando a entrada da rota 0.0.0.0.

```
Access-list 199 deny ip host 0.0.0.0 any
```

```
Access-list 199 permit ip any any
```

Pode-se utilizar palavras que representam redes, tais como host para um único endereço, ou any para qualquer endereço.

Comando: **Neighbor** *endereço IP* **distribute-list** *número da lista de acesso* **in/out**

É associado a uma lista de acesso ao vizinho, identificando se vai ser aplicado na entrada (recepção) ou na saída (divulgação).

No exemplo, a associação da lista de acesso 199, aplicado na entrada do vizinho 10.10.10.3

Neighbor 10.10.10.3 distribute-list 199 in

Altreando o exemplo inicial:

Comandos	Definição
Router bgp 100 Neighbor 172.17.10.1 remote-as 200 Neighbor 172.17.10.1 distribute-list 199 in Access-list 199 deny ip host 0.0.0.0 any Access-list 199 permit ip any any	Configura o vizinho 172.17.10.1 para não receber a rota 0.0.0.0.

Configuração de filtro Distribute-list

Filtro por prefix

Este filtro foi disponibilizado pela Cisco a partir da versão de IOS 12.0. Tem algumas vantagens, como melhoria de CPU devido a diminuição das listas processadas facilitando também a inserção e remoção em uma lista. No entanto, não serão utilizados estes tipos de experimentos no presente trabalho. Serão feitas as experiências usando os dois filtros vistos anteriormente, ou seja, por expressão AS-PATH, por distribute-list e o filtro por Route-maps, que será visto a seguir.

Filtro por route-maps

Route-map é o método usado para manipular informações de roteamento.

Funciona como uma lista de acesso avançada, muito utilizada no BGP para controlar ou modificar as informações de roteamento. Podem-se controlar os valores dos atributos.

Comandos route-map

Comando: Route-map *nome route-map [permit |deny] [número da seqüência]*.

O route-map tem várias sentenças organizadas com números que permitem as condições de seleção de rotas e regras para adequação dos valores relacionados aos seus atributos. Cada sentença possui o verbo *permit* ou *deny*.

No exemplo, a construção do route-map teste negando

Route-map teste deny 10

Comando: match *parâmetro e valor*

Comando: set *parâmetros e valor*

O parâmetro do comando match seleciona as rotas. O parâmetro do comando set define valores para os atributos.

No exemplo, apresentam-se todas as rotas da lista de acesso 199, e posteriormente ajusta-se o atributo da community para 50.

Math ip address 199

Set community 50

Comando: Neighbor *endereço IP* route-map *nome do route-map* {in/out}:

Associa o route-map no vizinho, definindo o seu sentido.

No exemplo, é vista a aplicação do route-map na entrada do vizinho 10.10.10.2

Neighbor 10.10.10.2 route-map teste in

Alterando o exemplo inicial:

Comandos	Definição
Router bgp 100 Neighbor 172.17.10.1 remote-as 200 Neighbor 172.17.10.1 route-map teste in Neighbor 172.17.10.1 send-community Route-map teste permit 10 Match as-path 10 Set community 50 Ip as-path access list 10 permit ^200\$	Nesse exemplo, em todas as rotas originadas no AS 200 o atributo community é ajustado para 50, ou seja, poderia ser útil para classificar essas rotas. A partir desta classificação, pode-se implementar uma política de roteamento.

Configuração por Route-Map

Obs: É necessário o comando (Neighbor *Endereço IP do vizinho* send-community) para enviar a community.

No decorrer deste trabalho, exemplificam-se alguns desses atributos. A seguir, apresenta-se a Tabela das possibilidades para o Match e o Set utilizadas na criação de route-maps.

Match as-path
Match ip address
Match ip route-source
Match route-type
Match community
Match ip next-hop
Match metric
Match tag

Parâmetros usados no Match

Os parâmetros usados com o comando set:

Set as-path
Set community
Set local preference
Set metric-type
Set origin
Set weight
Set ip next-hop
Set ip preference
Set level
Set metric
Set ip next-hop
Set tag

Parâmetros usados com o SET

Comandos de análise do roteamento

Existem comandos que permitem analisar o roteamento. Estes comandos podem ser aplicados no diagnóstico, monitoração e solução de eventuais problemas de roteamento.

Seguem os principais comandos utilizados nos experimentos do IXP, tanto nos roteadores Cisco usados pelos participantes, quanto nos servidores de roteamento Zebra.

Verificando as rotas da TABELA de roteamento

Verificar rota em uso

Comando: `show ip route rede`

Para verificar a rota que está em uso, ou seja, que está na Tabela de roteamento. Pode-se verificar através dele se um participante do PTT tem na sua Tabela de roteamento uma rota específica. Este comando trará um resultado distinto para cada roteador, pois trata a Tabela local de rotas.

Ex.: `show ip route 10.10.10.0` (Este comando mostrará a interface de destino para chegar na rede 10.10.10.0).

Verificar rota no BGP

Comando: `show ip bgp rede`

Para verificar a rota foi aprendida pelo protocolo BGP.

Ex.: `Show ip bgp 10.10.10.0` (Verificar se a rede 10.10.10.0 foi aprendida pelo protocolo BGP). As rotas apresentadas neste comando não estão necessariamente presentes na TABELA de roteamento, ou seja, precisam ter prioridades maiores do que outros protocolos para entrar na TABELA de roteamento.

Verificando os vizinhos e a quantidade de rotas recebidas

Comando: show ip bgp summary

O comando show ip bgp summary mostrará a relação dos vizinhos BGP, com as rotas recebidas.

Ex.: show ip bgp summary

Este comando além de exibir os vizinhos e quantidades de rotas recebidas. Pode-se observar o estado da máquina de estado do vizinho, ou seja, se o vizinho está em idle, connect, active, opensent, openconirm ou estabilished.

Verificando as rotas recebidas de um vizinho

Comando: sh ip bgp neighbor *endereço do vizinho* route

Este comando mostrará todas as rotas recebidas de um vizinho específico.

Ex.: sh ip bgp neighbor 10.10.10.1 route – Neste exemplo, mostram-se todas as redes recebidas de um vizinho específico. Pode ser usado para ver as redes recebidas em um servidor de roteamento e/ou um participante do PTT. Caso tenha filtro, o comando mostrará apenas as rotas selecionadas pelo filtro de entrada.

Verificando as rotas anunciadas por um vizinho

Comando: sh ip bgp neighbor *endereço do vizinho* advertised-routes

Este comando mostrará todas as rotas anunciadas por um vizinho específico.

Ex.: sh ip bgp neighbor 10.10.10.1 advertised-routes – Este exemplo mostrará todas as redes anunciadas por um vizinho específico. Pode ser usado para ver as redes anunciadas por um servidor de roteamento e/ou um participante do PTT. Caso

tenha filtro, o comando mostrará apenas as rotas que foram selecionadas pelo filtro de saída

Limpar a TABELA de roteamento do BGP

Comando: `clear ip bgp` *Endereço IP do Vizinho* ***in/out***

Este comando limpa as redes recebidas e anunciadas por este vizinho.

Ex.: `Clear ip bgp 10.10.10.2` – Neste exemplo as rotas recebidas e enviadas do vizinho 10.10.10.2 serão limpas. Neste comando pode-se restringir e limpar as rotas somente na entrada ou na saída.

ANEXO A – IXP (S) nos continentes

https://prefix.pch.net/applications/ixpdir/?show_active_only=0&sort=city&order=asc –
ACESSADO NOVEMBRO-2008 – 19-11

Region	Country	City	Internet Exchange Name	Participants	Traffic	Established	URL
Asia-Pacific	China	Beijing	CNC Beijing IXP	🌐	🌐	2008	🌐
Europe	France	Saint-Etienne	PhibIX Gix & IXP	2	🌐	2008	🌐
Africa	Cote D'Ivoire	Abidjan	Côte d'Ivoire Internet eXchange Point	5	4M	2006	🌐
Europe	United Arab Emirates	Abu Dhabi	Emirates Internet Exchange	🌐	🌐		🌐 🌐
		Abu Dhabi	Gulf Internet Exchange	🌐	🌐		🌐 🌐
Africa	Ghana	Accra	Ghana Internet Exchange	24	🌐	18 Oct 2005	🌐
Asia-Pacific	Australia	Adelaide	PIPE Networks Adelaide	18	🌐	2003	🌐
		Adelaide	South Australian Internet Exchange	🌐	🌐	Jul 1996	🌐
	Japan	Akita	Akita Regional IX	🌐	🌐		🌐 🌐
North America	United States	Albuquerque	IXNM	18	1.37G	Feb 2002	🌐
		Albuquerque	New Mexico Internet Exchange	🌐	🌐	2008	🌐
Europe	Kazakhstan	Almaty	Kazakhstan Internet Exchange (Kaz-IX), Almaty	🌐	🌐	2008	🌐
	France	Alsace	EuroGIX	🌐	🌐		🌐 🌐

	Netherlands	Amsterdam	Amsterdam Internet Exchange	305	574G	29 Dec 1997	
		Amsterdam	Netherlands Internet Exchange	197	30.6G	2002	
		Amsterdam	XchangePoint Amsterdam IPP				
North America	United States	Anchorage	Anchorage Metropolitan Access Point				
Africa	Tanzania	Arusha	Arusha Internet Exchange Point	6	400K	26 Jun 2006	
North America	United States	Ashburn	Equinix IBX Ashburn	72	80G	31 Oct 2002	
Latin America	Paraguay	Asuncion	CAPADI IXP-PY	15	40M	2000	
Europe	Greece	Athens	Athens Internet Exchange	15	8.74G	1997	
North America	United States	Atlanta	Atlanta Internet Exchange	43	7.03G	Aug 2003	
		Atlanta	MAE-Central				
		Atlanta	PAIX Atlanta				
		Atlanta	Atlanta IXP				
Asia-Pacific	New Zealand	Auckland	Auckland Peering Exchange	64		10 Aug 1999	
North America	United States	Austin	Austin Metro Access Point				
		Austin	AUSIX				
Asia-Pacific	Indonesia	Bali	Bali Internet Exchange				
North America	United States	Baltimore	Baltimore-IXP				
Asia-Pacific	Indonesia	Bandung	NPXP				
	Thailand	Bangkok	CAT National Internet Exchange	20	1.7G		
		Bangkok	NECTEC-IIR				
		Bangkok	ThaiSarn Public Internet Exchange				
Europe	Spain	Barcelona	Catalunya Neutral Internet Exchange Point	19	614M		
Asia-Pacific	China	Beijing	Beijing National Internet Exchange Center	11	2G	Apr 2000	
		Beijing	TerreIXP Beijing				
Europe	Lebanon	Beirut	Beirut Internet Exchange	4		6 Dec 2007	
Latin America	Brazil	Belo Horizonte	Ponto de Troca de Tráfego do Belo Horizonte	3	2.24M		

North America	United States	Bend	Central Oregon Exchange			
		Berkeley	Calren2 North			
		Berkeley	6TAP			
Europe	Germany	Berlin	Berlin Commercial Internet Exchange	42	2.39G	2008
		Berlin	Berlin Internet Exchange/European Commercial Internet Exchange	16		2003
	Spain	Bilbao	Punto Neutro Vasco de Internet	7	82.1M	2002
North America	United States	Billings	Yellowstone Regional Internet Exchange	4		2003
Latin America	Colombia	Bogotá	IXP-Colombia	13	3.2G	29 Jun 2000
		Bogota	IXP of the Americas			Mar 2008
North America	United States	Boston	Boston Metropolitan Exchange Point	17	17.4M	
		Boston	PAIX-Boston			
Latin America	Brazil	Brasília	Ponto de Troca de Tráfego do Brasília	6	273M	
Europe	Slovakia	Bratislava	SitelIX	7	281M	
		Bratislava	Slovak Internet Exchange	47	16.4G	Oct 1996
Asia-Pacific	Australia	Brisbane	PIPE Networks Brisbane	49		May 2002
		Brisbane	AusBONE-Brisbane Internet Exchange			Oct 1997
Europe	Czech Republic	Brno	Commercial Brno Internet Exchange			12 May 2007
		Brno	Neutral Internet Exchange			
	Belgium	Brussels	Belgian National Internet Exchange	46	16.6G	1995
		Brussels	Free Belgian Internet Exchange	34		4 Sep 2003
	Romania	Bucharest	InterLAN-IX	47	13.3G	2007
		Bucharest	Romanian National Internet Exchange	34	6.3G	7 Sep 2001
		Bucharest	Bucharest Internet Exchange			1999
	Hungary	Budapest	Budapest Internet Exchange	52	98.5G	1996

Latin America	Argentina	Buenos Aires	IXP CABASE	79	1.42G	Apr 1998	
		Buenos Aires	Cubecorp				
Africa	Burundi	Bujumbura	Burundi Internet Exchange Point				
	Egypt	Cairo	Cairo Internet Exchange	3	12M	Jul 2005	
		Cairo	Middle East Internet eXchange	7		May 2007	
		Cairo	Cairo Regional Internet Exchange			May 2002	
Asia-Pacific	Australia	Canberra	ACT Internet Exchange	8	1G	1 Oct 2008	
		Canberra	PIPE Networks Canberra	4		Jun 2004	
Africa	South Africa	Cape Town	Cape Town Internet Exchange				
		Cape Town	Hub				
Europe	Russia	Chelyabinsk	Chelyabinsk Peering Point	12	3M		
Asia-Pacific	India	Chennai	National Internet Exchange of India	9	85M		
North America	United States	Chicago	Ameritech Advanced Data Services Network Access Point				
		Chicago	Any2 Exchange				
		Chicago	Chicago Internet Exchange	12		11 Apr 2008	
		Chicago	Equinix Chicago IBX	36		4 Mar 2002	
		Chicago	MAE-Chicago				
		Chicago	PacBell ATM Exchange				
		Chicago	STAR TAP	30		1998	
		Chicago	StarLight	34		Mar 2001	
Asia-Pacific	New Zealand	Christchurch	Christchurch Internet Exchange	7		2007	
North America	United States	Colchester	VermontIX				
Asia-Pacific	Sri Lanka	Colombo	Sri Lanka Internet Exchange	20		1 Jul 2001	
		Colombo	Singapore Telecom Internet Exchange				
North America	United States	Columbus	CMH-IX				
		Columbus	Columbus Internet Exchange				
		Columbus	FiberIXP				
Europe	Denmark	Copenhagen	Danish Internet Exchange	46	13G		

Latin America	Netherlands Antilles	Curaçao	Caribbean Internet Exchange				
	Brazil	Curitiba	Ponto de Troca de Tráfego do Paraná	15	1.03G	Jun 2002	
Asia-Pacific	Viet Nam	Da Nang	Viet Nam Internet Exchange				
North America	United States	Dallas	Dallas/Fort Worth Metropolitan Access Point				
		Dallas	Equinix IBX Dallas			15 Jun 2001	
		Dallas	MAE Central				
		Dallas	PAIX-Dallas			24 Feb 2002	
		Dallas	Excel Telecom IXP				
		Dallas	MAE-Dallas				
Africa	Tanzania	Dar es Salaam	Tanzania Internet Exchange	17	1.01M	Jul 2003	
Europe	Germany	Darmstadt	Metropolitan Area Network Darmstadt				
North America	United States	Dayton	Dayton Metro Internet Exchange	4		24 Jun 2003	
Europe	Netherlands	Den Haag	Den Haag Internet Exchange			1997	
North America	United States	Denver	Mountain Area Exchange				
		Denver	Denver Internet Exchange				
		Denver	PAIX-Denver				
Asia-Pacific	Bangladesh	Dhaka	Bangladesh Internet Exchange	24	21M	26 Aug 2004	
		Dhaka	Bangladesh Society of Internet Exchange			May 2004	
Europe	Ukraine	Donetsk	East Ukrainian Internet Exchange	14			
	Ireland	Dublin	Internet Neutral Exchange Association Limited	36	2.85G	1996	
Asia-Pacific	New Zealand	Dunedin	Dunedin Peering Exchange	3		2005	
Europe	Germany	Düsseldorf	Berlin Internet Exchange/European Commercial Internet Exchange	46	5.85G	2003	
		Edinburgh	Scottish Internet Exchange			Sep 1999	
		Edinburgh	WorldIX	4		Feb 2000	
	United Kingdom	Edinburgh	UNION IXP				

North America	Canada	Edmonton	Edmonton Internet Exchange			
Europe	Russia	Ekaterinburg	Ekaterinburg Internet Exchange			
	Netherlands	Enschede	Nederlands-Duitse Internet Exchange	21	50M	Apr 2001
North America	United States	Eugene	Oregon Internet Exchange			
Europe	Italy	Florence	Tuscany Internet Exchange	24	194M	
Latin America	Brazil	Florianópolis	Ponto de Troca de Tráfego do Florianópolis	45	164M	
Europe	Germany	Frankfurt	Deutscher Commercial Internet Exchange	235	548G	May 1995
		Frankfurt	KleyReX Internet Exchange	102	1.46G	Sep 2002
		Frankfurt	MAE-Frankfurt			
		Frankfurt	Frankfurt Network Access Point			
		Frankfurt	XchangePoint Frankfurt IPP			
Asia-Pacific	Japan	Fukuoka	BBIX			
Africa	Botswana	Gaborone	Botswana Internet Exchange	8		
Europe	Switzerland	Geneva	CERN Internet Exchange Point	23	1.35G	
	Sweden	Gothenburg	Gothenburg Internet Exchange	12		
		Gothenburg	Netnod Gothenburg	14	12G	
Africa	South Africa	Grahamstown	Grahamstown Internet Exchange	6	3M	13 Mar 2005
Europe	Austria	Graz	Grazer Internet Exchange			
	Netherlands	Groningen	Groningen-IX	66		2001
Asia-Pacific	China	Guangzhou	not available			Dec 2000
Latin America	Ecuador	Guayaquil	IXP AEPROVI Guayaquil	9	15M	2003
Europe	Sweden	Gävle	Gästrikland-Hälsingland Regional Internet Exchange	14	398M	
Latin America	Cuba	Habana	IXP de Cuba	5	50M	Jun 2001
Europe	Germany	Hamburg	Internet Exchange Hamburg			Sep 2002
		Hamburg	Work-IX Hamburg	30	1.72G	2008
		Hamburg	XchangePoint Hamburg IPP	2		Apr 2003

		Hamburg	Hansestadt Hamburg Commercial Internet Exchange				
Asia-Pacific	Viet Nam	Hanoi	Viet Nam Internet Exchange		140M	8 Nov 2003	
Africa	Zimbabwe	Harare	Zimbabwe Internet Exchange	5		Jul 2001	
North America	United States	Helena	Northwest Internet Exchange	2		Feb 2006	
Europe	Finland	Helsinki	Finnish Communication and Internet Exchange	27	23.1G	1993	
Asia-Pacific	Viet Nam	Ho Chi Minh City	Viet Nam Internet Exchange		160M	Mar 2004	
	Australia	Hobart	PIPE Networks Hobart	6		Aug 2003	
		Hobart	Tasmanian Internet Exchange				
	Hong Kong	Hong Kong	Equinix Hong Kong				
		Hong Kong	Hong Kong Internet Exchange	78	66G	Apr 1995	
		Hong Kong	Hong Kong Academic Internet Exchange				
		Hong Kong	Level3				
		Hong Kong	Pihana Hong Kong				
North America	United States	Honolulu	DRFExchange			8 Jan 2007	
		Honolulu	Hawaii Internet Exchange	10			
		Houston	Compaq Houston IXP				
		Houston	Houston Metropolitan Access Point				
		Houston	Rice EduPOP			10 May 2001	
		Houston	Texas GigaPOP				
		Houston	Houston MAGIE				
		Houston	MAE-Houston				
		Houston	TexasSEGP				
Africa	Nigeria	Ibadan	Ibadan Internet Exchange	2		Mar 2002	
North America	United States	IndiaIXPolis	IndiaIXPolis Data Exchange				
Asia-Pacific	New Zealand	Invercargill	Southland Internet Exchange				
Europe	Turkey	Istanbul	Turkish Internet Exchange			1997	
Asia-Pacific	Indonesia	Jakarta	Biznet Internet Exchange	18	76.8M		
		Jakarta	IndonesiaEP				

		Jakarta	Indonesian Internet Exchange	99	3.4G	Aug 1997	
		Jakarta	IXPSINDO International Internet Exchange				
		Jakarta	Jakarta Internet Exchange				
		Jakarta	PT. Cyberindo Aditama			2008	
		Jogjakarta	Indonesian Internet Exchange			Mar 2004	
Africa	South Africa	Johannesburg	Johannesburg Internet Exchange	26	238M	6 Jun 1996	
		Johannesburg	South African Internet Exchange				
	Uganda	Kampala	Uganda Internet Exchange	5	3.2M	Jul 2003	
North America	United States	Kansas City	Kansas City Internet Exchange	12		2006	
Asia-Pacific	Pakistan	Karachi	Pakistan Network Access Point				
	Nepal	Kathmandu	Nepal Internet Exchange	13	36M	Aug 2002	
Europe	Poland	Katowice	Tysiaclecie Internet Exchange	3			
	Ukraine	Kharkov	Odessa Internet Exchange	12	1.05G	Sep 2007	
		Kherson	Kherson Internet Exchange	5		2008	
		Kiev	Ukrainian Internet Exchange	89	33.1G	Jul 2000	
		Kiev	Ukrtelecom Internet Exchange				
Africa	Rwanda	Kigali	Rwanda Internet exchange				
	Congo-Kinshasa	Kinshasa	Kinshasa Internet Exchange	4		Nov 2002	
Asia-Pacific	India	Kolkotta	National Internet Exchange of India	4	500K		
Europe	Russia	Krasnoyarsk	Krasnoyarsk Internet Exchange	35			
Asia-Pacific	Malaysia	Kuala Lumpur	Malaysia Internet Exchange	23	838M	15 Dec 2006	
		Kuala Lumpur	Asia Regional Internet Exchange IXP				
		Kuala Lumpur	Kuala Lumpur Internet Exchange				
	Japan	Kyoto	Kyoto Municipal Exchange				
Latin America	Bolivia	La Paz	IX de Bolivia				
Africa	Nigeria	Lagos	Internet eXchange Point of Nigeria	18		May 2007	

Europe	Netherlands	Leeuwarden	Frieze Internet Exchange	23	3.45G	26 Jan 2006	
	Cyprus	Lefkosia	Cyprus Internet Exchange	5	10M		
	Germany	Leipzig	European Commercial Internet Exchange				
Latin America	Peru	Lima	IXP Lima				
		Lima	IXP Perú			14 May 2001	
Europe	Sweden	Linköping	Linköping municipal exchange			Oct 2002	
	Portugal	Lisbon	Gigabit Portuguese Internet Exchange	20	6.01G	1997	
Asia-Pacific	Australia	Lismore	lismore internet exchange				
Europe	Slovenia	Ljubljana	Slovenian Internet Exchange	16			
	United Kingdom	London	LoIXP/XchangePoint shared interconnect	56		2004	
		London	London Internet Exchange	298	362G	8 Nov 1994	
		London	London Internet Providers Exchange	55	152M	Oct 2001	
		London	London Network Access Point	88	8.8G	1998	
		London	Meridian Gate Internet Exchange	20			
		London	Redbus Internet Exchange	15		13 Apr 2004	
		London	XchangePoint London IPP	166	5.3G	2001	
		London	EarthIXP			26 Feb 2007	
		London	SovEx				
		London	UK6X				
	United States	Los Angeles	6IIX				
		Los Angeles	Calren2 South				
		Los Angeles	Equinix Los Angeles IBX			11 Jul 2002	
		Los Angeles	Los Angeles International Internet Exchange	46	5.54G	1996	
		Los Angeles	MAE-LA				
		Los Angeles	One Wilshire Any2 Exchange	170	6G	Dec 2005	
		Los Angeles	Pac Bell Los Angeles Network Access Point				
		Los Angeles	Equinix Los Angeles IBX				
		Los Angeles	PAIX-LA				

		Los Angeles	Pihana Los Angeles				
		Louisville	Louisville IXP				
Africa	Angola	Luanda	Angola Internet Exchange	8	3M	17 Mar 2006	
		Luanda	Ponto de Intercambio Internet Angola				
Europe	Sweden	Luleå	Netnod Luleå	6	100M	Sep 2004	
		Luleå	PolarIX			Sep 2002	
Africa	Zambia	Lusaka	Zambia Internet Exchange Point				
Europe	Luxembourg	Luxembourg	Luxembourg Internet Exchange	11	45.8M		
	France	Lyon	Lyon Internet Exchange	22			
North America	United States	Madison	Madison Internet Exchange	12		2000	
Europe	Spain	Madrid	Espana Internet Exchange	48	116G	1997	
		Madrid	Madrid Internet Exchange	9	16M		
		Madrid	Punto Neutro Español de Internet				
		Madrid	IXP de las Madrid				
	Sweden	Malmö	Malmö Internet eXchange	13			
		Malmö	Netnod Malmö	16	22.8G		
Latin America	Nicaragua	Managua	Nicaraguan Internet Exchange	10		Apr 2004	
Europe	United Kingdom	Manchester	Manchester Commercial Internet Exchange	14	400M		
		Manchester	Manchester Network Access Point	51	1.8G	1997	
		Manchester	Enlightened Internet Exchange			4 Jun 2005	
Asia-Pacific	Philippines	Manila	Common Routing Exchange	8			
		Manila	Philippine Open Internet Exchange	10	5M	2007	
		Manila	Manila Internet Exchange				
		Manila	Philippines Internet Exchange				
Africa	Mozambique	Maputo	Mozambique Internet Exchange	8	5M	Jul 2002	
North America	United States	Marina del Rey	ISI DMZ/Gigapop				

Europe	France	Marseille	Marseille Internet Exchange	2	128K	2005	
Africa	Swaziland	Mbabane	Swaziland Internet Exchange	2	128K	Jun 2004	
Asia-Pacific	Australia	Melbourne	PIPE Networks Melbourne	35		Jul 2003	
		Melbourne	Victorian Internet Exchange	17		1998	
		Melbourne	AusBONE-Melbourne Internet Exchange			1997	
		Melbourne	AUSIX			Jul 2001	
		Melbourne	Melbourne IXPette			Apr 1999	
		Melbourne	Melbourne IXPette			Apr 1999	
North America	United States	Miami	AMPATH - Florida International University/CIARA	4		2006	
		Miami	IXP of the Americas	80	20G	12 Aug 2004	
		Miami	BellSouth Multimedia Internet Exchange				
		Miami	PAIX-Miami				
Europe	Italy	Milan	Milan Internet Exchange	70	27.3G	Nov 2000	
North America	United States	Minneapolis	Northern Lights GigaPoP			2000	
		Missoula	Northwest Internet Exchange	2		15 Dec 2004	
		Moffett Field	AMES-MIX				
		Moffett Field	MAE-West				
		Moffett Field	NGIX-West				
	Canada	Montreal	Quebec Internet Exchange/RISQ				
		Montreal	Quebec Internet Exchange/RISQ				
Europe	Russia	Moscow	Moscow Internet Exchange	243	50.3G	1995	
	Malta	Msida	Malta Internet Exchange	12		7 Sep 1999	
Asia-Pacific	India	Mumbai	National Internet Exchange of India	20	500M		
		Mumbai	Enron India Exchange				
Europe	Germany	Munich	Internet Exchange Service	34	4.1G	Jun 1994	
		Munich	Munich Commercial Internet Exchange				
		Münster	Nederlands-Duitse Internet Exchange				
Asia-Pacific	Japan	Nagoya	BBIX				
		Nagoya	JPIX Nagoya	9	64G	Apr 2001	
Africa	Kenya	Nairobi	Kenya Internet Exchange Point	25	69.8M	Feb 2001	

North America	United States	Nashville	Nashville IXP				
		Nashville	Nashville City Net				
Asia-Pacific	India	New Delhi	National Internet Exchange of India	17	100M		
		New Delhi	India Internet Exchange				
North America	United States	New Jersey	MAGPI Gigapop				
		New York	6IIX				
		New York	Big Apple Peering Exchange (formerly NY6iX)	29			
		New York	Equinix New York IBX			11 Mar 2003	
		New York	MAE-NY				
		New York	MetrolIX				
		New York	New York International Internet Exchange	110	61.4G	1996	
		New York	PAIX-NY				
		New York	Peer 1 Internet Exchange			9 Jan 2004	
		New York	New York City Exchange				
		New York	NY-RIX				
Asia-Pacific	Japan	Nihonbashi	JPIX Nihonbashi			Apr 2004	
Europe	Russia	Novosibirsk	Novosibirsk Internet Exchange	35			
	Germany	Nuremberg	Nürnberg Internet Exchange	22	1.98G		
North America	United States	Ohio	OhioEP				
Europe	Portugal	Oporto	Gigabit Portugese Internet Exchange				
Asia-Pacific	Japan	Osaka	BBIX				
		Osaka	JPIX Osaka	10		Apr 2002	
		Osaka	JPIX Osaka	23	43G	2002	
		Osaka	NSPIX-3	23	72G	Mar 1997	
		Osaka	NSPIX-2			Mar 1997	
Europe	Norway	Oslo	Free Internet eXchange Oslo	5		16 Jan 2008	
		Oslo	Norwegian Internet Exchange	65	19.2G	30 Aug 2000	
North America	Canada	Ottawa	Ottawa Internet Exchange	14	190M	10 Apr 2001	

Asia-Pacific	New Zealand	Palmerston North	Palmerston North Internet Exchange	4		30 Nov 2004	
North America	United States	Palo Alto	PAIX		41G	Dec 1994	
Latin America	Panama	Panama City	Intered Panamá	10			
		Panama City	IXP Panamericano S.A.				
Europe	France	Paris	Free-IX	106			
		Paris	French National Internet Exchange IPv6	17		1 Nov 2002	
		Paris	Gigabit European Internet eXchange	8			
		Paris	MAE-Paris	18			
		Paris	Mix Internet Exchange and Transit	7			
		Paris	Paris Internet Exchange Service				
		Paris	Paris IXP	124	65.7G	Jul 2005	
		Paris	Paris Operators for Universal Internet eXchange	30			
		Paris	PARIX	46	14G	2001	
		Paris	Service for French Internet Exchange	84		1995	
		Paris	Paris Internet Exchange				
North America	United States	Pennsauken	Sprint IXP				
Asia-Pacific	Australia	Perth	Western Australia Internet Exchange	56	2.1G	1997	
North America	United States	Philadelphia	PAIX-Philly				
		Philadelphia	Philadelphia Internet Exchange				
Latin America	Netherlands Antilles	Philipsburg	Open Caribbean Internet eXchange			Oct 2008	
Asia-Pacific	Cambodia	Phnom Penh	Cambodia Internet Exchange	4	3M	Jun 2003	
North America	United States	Phoenix	Phoenix Internet Exchange				
		Phoenix	TelX Internet Exchange			2 Jun 2008	
		Phoenix	PhoenixIXP				

		Pittsburgh	NCNE GigaPOP				
		Pittsburgh	PittsburghEP	7	8.5M		
Latin America	Haiti	Port au Prince	Haiti Internet Exchange Point	3		Mar 2008	
Africa	Mauritius	Port Louis	Mauritius Internet Exchange	4		Dec 2005	
North America	United States	Portland	Northwest Access Exchange	22		Dec 2002	
		Portland	MaineXchangePoint				
Latin America	Brazil	Porto Alegre	Rio Grande do Sul Internet Exchange	19	770M	20 Nov 2000	
Europe	Poland	Poznan	Poznan Internet Exchange	15			
	Czech Republic	Prague	Neutral czFree eXchange	12	2.97G		
		Prague	Neutral Internet Exchange	80	62G	Aug 1996	
Latin America	Ecuador	Quito	IXP AEPROVI Quito	6	25M	2003	
Europe	Iceland	Reykjavik	Reykjavik Internet Exchange	15	1.58G		
North America	United States	Richardson	North Texas GigaPOP	8		20 Aug 2004	
Europe	Latvia	Riga	Latvian GIX-RIX	142			
Latin America	Brazil	Rio de Janeiro	Ponto de Troca de Tráfego do Rio de Janeiro	6	454M		
		Rio De Janeiro	Optiglobe Internet Exchange - Latin America				
Europe	Saudi Arabia	Riyadh	Saudi Telecommunications Company Internet NOC				
		Riyadh	The Internet Services Unit			1998	
	Italy	Rome	Nautilus Mediterranean Exchange Point	28	9.9G	1996	
		Rome	CASPUR				
	Netherlands	Rotterdam	Rotterdam Internet eXchange				
North America	United States	Salt Lake City	Utah Regional Exchange Point	13	9M	23 Apr 1996	
Europe	Russia	Samara	Samara Internet Exchange	5			
North America	United States	San Antonio	San Antonio Metro Access Point				
		San Diego	San Diego Network Access Point	21		Feb 1998	

		San Diego	PAIX-SD				
		San Francisco	PAIX San Francisco				
		San Francisco	PB-IXP				
		San Francisco	San Francisco Metropolitan Internet Exchange	10		May 2006	
		San Jose	Equinix IBX San Jose	37		25 May 2000	
		San Jose	MAE-West-ATM				
		San Jose	KDWEB Exchange				
	Puerto Rico	San Juan	Internet Exchange of Puerto Rico	2		16 Apr 2006	
Latin America	El Salvador	San Salvador	IXP El Salvador				
North America	United States	Santa Clara	6IIX				
Latin America	Chile	Santiago	IXP Chile	12			
Europe	Spain	Santiago de Compostela	Punto Neutro de Intercambio de Tráfico de Internet en Galicia	6	65M	19 Jul 2002	
Latin America	Dominican Republic	Santo Domingo	IXP del Caribe			Jul 2008	
	Brazil	São Paulo	IXP do Brasil	37	9.6G	1998	
		São Paulo	Ponto de Troca de Tráfego Metro	50	6.7G	4 Oct 2004	
		São Paulo	Telcomp	5	75M	2003	
		São Paulo	Tivit	39			
		São Paulo	Daedalus				
		São Paulo	Diveo IXP				
		São Paulo	IXP Abranet (Not an exchange)				
North America	United States	Seattle	PAIX-Seattle				
		Seattle	Seattle Internet Exchange	117	18.5G	1997	
		Seattle	Seattle Northwest Network Access Point				
		Seattle	NIX				

Asia-Pacific	Japan	Seattle	NWnet Gigapop
		Secaucus	Equinix IBX Secaucus
	Korea	Sendai	BBIX
		Seoul	6NGIX	8
		Seoul	DACOM Internet Exchange	52	70G	Oct 1996
		Seoul	Korea Internet Data Center	200	300G	Oct 1999
		Seoul	Korea Internet Exchange KINX	39	49.9G	Jun 1999
		Seoul	Korea Internet Exchange KIX	15	6G	Oct 1996
		Seoul	Korea Telecom Internet Exchange			Oct 1996
		Seoul	Pihana Seoul
	China	Shanghai	Asia Pacific Internet Exchange
		Shanghai	Shanghai City Exchange
		Shanghai	Shanghai Internet Exchange			Dec 2000
	Singapore	Singapore	Equinix Singapore IBX
		Singapore	Singapore Open Exchange	18		Aug 2001
		Singapore	Pihana Singapore
		Singapore	Singapore One Network for Everyone
		Singapore	Singapore Telecom Internet Exchange
Europe	Bulgaria	Sofia	GoCIS
		Sofia	Sofia Internet Exchange
North America	United States	Spokane	Eastern Washington Internet Exchange
		St. Louis	St. Louis Regional Exchange Point	20		21 Feb 2006
		St. Louis	St. Louis Internet Access Consortium
		St. Louis	StLouisIXP
		St. Louis	StLoulX
Europe	Russia	St. Petersburg	MPIX	5
		St. Petersburg	St. Petersburg Internet Exchange	28
		St. Petersburg	NW-IX

	Sweden	Stockholm	Netnod Stockholm	48	108G	1997		
		Stockholm	SOL-IX	22	1G	Dec 2001		
		Stockholm	Stockholm Internet Exchange	20	281M	2007		
	Germany	Stuttgart	Stuttgarter Internet Exchange	9		Apr 2005		
	Romania	Suceava	Suceava Internet Exchange					
	Sweden	Sundsvall	Netnod Sundsvall	14	11.4G			
	Asia-Pacific	Indonesia	Surabaya	Indonesian Internet Exchange			Oct 2003	
			Surabaya	NPXP				
		Australia	Sydney	Equinix Sydney	18	108M	2003	
			Sydney	PIPE Networks Sydney	81		2003	
		Sydney	AusBONE-Sydney Internet Exchange			Oct 1997		
		Sydney	AUSIX.net					
		Sydney	Pihana Sydney					
		Taiwan	Taipei	Taipei Internet Exchange	13	60M	2 Jul 2002	
			Taipei	Taiwan Internet Exchange	39	2.73G	Dec 1997	
			Taipei	Taiwan Network Access Point	16	1G	2001	
			Taipei	TaiwanEP				
			Taipei	not available				
			Taipei	TaipeiGigaPoP				
	North America	United States	Takoma Park	Andrew's Basement Exchange				
Europe	Estonia	Tallinn	Linx Telecom Tallinn IX	9	720M			
		Tallinn	Tallinn Internet Exchange	13	2.91G	2002		
North America	United States	Tampa	Tampa Internet Exchange	4		16 May 2005		
		Tampa	Tampa Internet Exchange					
Europe	Finland	Tampere	Tampere Region Exchange	5		3 Aug 2003		
	Israel	Tel Aviv	Israeli Internet Exchange	13	1.32G	1996		
Asia-Pacific	Japan	Tokyo	BBIX					
		Tokyo	Equinix IBX Tokyo					
		Tokyo	Japan Internet Exchange	113	108G	Jul 1997		
		Tokyo	JPIX Tokyo I, Otemachi	52	171G	2001		
		Tokyo	JPIX Tokyo II, Ikebukuro			1 Apr 2008		
		Tokyo	Network Service Provider IXP	85	22.2G	Oct 1996		

			(Distributed IX in Edo)				
		Tokyo	Network Service Provider IXP-6	51	100M	Aug 1999
		Tokyo	Media Exchange Co.
		Tokyo	Network Service Provider IXP			Apr 1994
		Tokyo	Pihana Tokyo
Europe	Italy	Torino	Torino Piemonte Exchange Point	38	10.8G	2002
North America	Canada	Toronto	CANIX
		Toronto	Toronto Internet Exchange	75	11.1G	1998
		Toronto	Peer1 Internet Exchange - Toronto			2008
	United States	Tucson	Tucson Interconnect
		Tucson	Tucson Network Access Point
Asia-Pacific	Mongolia	Ulaanbaatar	Mongolia Internet Exchange	6
Europe	Russia	Ulyanovsk	Simbirsk Internet Exchange
	Sweden	Umeå	NorrNod	10	150M	1996
Asia-Pacific	New Zealand	Upper Hutt/Lower Hutt/Porirua	Three Cities Internet Exchange	4		Jun 2005
North America	Canada	Vancouver	British Columbia Internet Exchange
		Vancouver	Peer1 Internet Exchange - Vancouver			2008
Europe	Bulgaria	Varna	Varna Internet Exchange
	Austria	Vienna	Vienna Internet Exchange	105	21.6G	Apr 1996
North America	United States	Vienna	PAIX-VA
Europe	Lithuania	Vilnius	TIX	24		2008
Asia-Pacific	New Zealand	Waikato	New Zealand Internet Exchange (Defunct)
Europe	Poland	Warsaw	ATMAN AC-X Internet Exchange
		Warsaw	Polish Internet Exchange	77	39.6G	2 Apr 2006
		Warsaw	Warsaw Internet Exchange	43	3.87G	1996
North	United	Washington,	MAE-East			Dec 1993

America	States	D.C.				
		Washington, D.C.	MAE-East-ATM			Jul 1998
		Washington, D.C.	NeutralIXP			
Asia-Pacific	New Zealand	Wellington	Wellington Internet Exchange	181	540M	Dec 1996
Europe	Croatia	Zagreb	Croatian Internet Exchange	18	238M	
	Ukraine	Zaporozhye	Zaporozhye Internet Exchange			2008
	Switzerland	Zurich	SwissIX	89	3.11G	
		Zurich	The Internet Exchange	65	5.58G	1999
		Zurich	MAE-Zurich			
		Zurich	Swiss Internet Exchange			
		Zurich	Telehouse Internet Exchange			1999
Latin America	Chile		Chile Network Access Point			
	Colombia		CIN			
	Panama		Secretaria Nacional de Ciencia, Tecnologia e Innovacion			
Europe	Poland		Krakowski Internet eXchange			
			Wroclaw Internet eXchange			
	Ukraine		Leontovich 9 IX			2008