PORTAL DE ANÁLISE DE TRÁFEGO

Por

Guilherme Silva Vilela

PROJETO FINAL DE CURSO
UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
RIO DE JANEIRO, RJ
AGOSTO 2003

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO

ESCOLA DE ENGENHARIA

DEPARTAMENTO DE ELETRÔNICA

PORTAL DE ANÁLISE DE TRÁFEGO

Autor: Guilherme Silva Vilela

Orientador: Luis Felipe M. de Moraes, Ph.D.

Co-Orientador: Claus Rugani Töpke, MsC.

Examinador: Aloysio de Castro P. Pedroza, Dr.Ing.

Examinador: Marcelo Luiz Drumond Lanza, MsC.

DEL

Agosto de 2003

A minha família

A minha namorada Rafaela

Agradecimentos:

Ao meu Orientador Luis Felipe M. de Moraes por todo apoio dado a este projeto.

Ao amigo Claus pela grande ajuda.

A toda equipe do Ravel e a todas as pessoas que ajudaram e contribuíram neste projeto.

A equipe da Coordenação de Engenharia de Operações (CEO)/Rede-Rio pelo importante apoio prestado na coleta dos dados.

Resumo

Um dos maiores desafios para backbones IP é identificar qual tráfego passa pela rede e como ele passa. Outro grande problema encontrado é que as informações obtidas não são compartilhadas, e muitas vezes torna-se difícil caracterizar o tráfego de uma forma geral e abrangente. O objetivo deste trabalho é proporcionar uma maior divulgação sobre este tema, criando um portal de análise de tráfego contendo diversas informações sobre o assunto, além de disponibilizar todos os estudos realizados na RedeRio, contribuindo para um maior entendimento sobre o tráfego em backbones IP.

Palavras-Chave

Caracterização de Tráfego

Backbone IP

Internet

Fluxos de Comunicação

Portal

Sumário

Lista de Figuras

Lista de Tabelas

Abreviaturas e Siglas

AS - *Autonomous System*

CAIDA - *Cooperative Association for Internet Data Analysis*

IP - *Internet Protocol*

RRD - *Round Robin Database*

TCP - *Transmission Control Protocol*

UDP - *User Datagram Protocol*

Capítulo 1

Introdução

1.1 Motivação

Desde o surgimento da ARPANET, oferecendo inicialmente o serviço de endereço eletrônico, e posteriormente serviços como FTP e Telnet, a Internet vem crescendo em número de usuários e diversidade de serviços [11].

Hoje a Internet serve a propósitos comerciais e sociais para um grande número de pessoas. Sendo tecnologicamente muito diversa tanto em infra-estrutura de rede como em protocolos e aplicações que a utilizam, a Internet continua a se desenvolver como uma infra-estrutura global para novos serviços, como multimídia *streaming*. Essa diversidade torna os parâmetros que definem o desempenho de um backbone IP e que caracterizam seu tráfego bastante complexos [1]. Consequentemente, provedores tem se motivado cada vez mais a ganhar um conhecimento mais profundo do comportamento da Internet, através de medidas de tráfego e de desempenho de rede [12]. Neste aspecto é interessante saber aonde o tráfego se inicia, qual é o seu destino e quais são as aplicações mais utilizadas. A análise dessas medidas é útil para diversos propósitos como por exemplo para verificação de garantia de serviço, para cobranças, gerenciamento de recursos, engenharia de tráfego e planejamento de rede. No entanto a questão de medição de desempenho vem recebendo pouca atenção, apesar de ser importante para que se possa ter medidas qualitativas e quantitativas do tráfego, tornando possível que um backbone IP tenha uma avaliação de seu funcionamento e como ele deve prosseguir no futuro [3].

Apesar dos provedores monitorarem suas redes, a natureza competitiva do mercado de serviços da Internet vem desencorajando uma cooperação global para permitir uma medida

em larga escala de performance da Internet. A cooperação é necessária para garantir que qualquer instrumentação e método utilizado para monitorar a Internet, seja consistente, acurado, escalonável e seguro [11]. É nesse contexto que esse trabalho se enquadra, fazendo uma divulgação de informações sobre a análise de tráfego.

1.2 Objetivos

As definições feitas pela comunidade de redes de computadores, sobre medidas de desempenho e caracterização de tráfego IP na Internet, são muitas vezes precárias e limitadas [1] [3] devido a sua complexidade. Este trabalho tem como objetivo a criação de um portal contendo diversos documentos e links sobre o assunto, fazendo do portal, uma poderosa fonte de consulta para a comunidade científica e para as empresas que utilizam backbones IP, no que diz respeito à caracterização de tráfego, tanto nas suas redes, quanto em outras que possam ser de interesse.

O trabalho tem como diferencial o fato de poder tornar pública toda pesquisa feita, a descrição da infra-estrutura montada para a coleta e análise dos dados e as diversas medidas e estudos efetuados na RedeRio, contribuindo para uma maior divulgação e entendimento da caracterização de tráfego em backbones IP.

A infra-estrutura montada visa a implementação de ferramentas para:

- Coleta de dados relacionados a parâmetros diversos (latência, jitter, perda de pacotes, vazão, retransmissão, etc.), em backbones IP.

- Tratamento dos dados obtidos para, através de uma interface amigável, apresentar aos usuários o sistema implementado. Em particular, o material desenvolvido será aplicado à RedeRio/FAPERJ, podendo também ser implementado em outros backbones, de quem venha manifestar interesse, de forma a funcionar como uma referência nacional sobre o assunto.

1.3 Estrutura do Texto

Neste capítulo são apresentas as motivações e objetivos deste projeto, demonstrando sua importância e contribuição.

No capítulo dois é apresentado o ambiente para a coleta de dados, incluindo a metodologia utilizada na coleta e explicitando como cada etapa foi efetuada. Em seguida é feita uma descrição do formato dos dados coletados e finalmente é feita uma descrição detalhada das ferramentas utilizadas para implementação da coleta e análise dos dados obtidos.

No capítulo três é mostrada a configuração do coletor e os principais aplicativos instalados. Em seguida no capítulo quatro, é feita uma descrição do portal.

No capítulo cinco os dados obtidos da RedeRio são apresentados, fazendo algumas análises sobre esses dados. Finalmente no capítulo seis é feita uma conclusão do trabalho efetuado e sugestões para trabalhos futuros.

Capítulo 2

Ambiente para Coleta

Para fazer a análise do tráfego é necessário ter um ambiente adeguado. Neste capítulo será feita uma abortagem sobre a metodologia utilizada para a coleta, quais tipos de dados foram coletados e as ferramentas utilizadas.

2.1 Metodologia

Nesta seção será feita uma revisão da metodologia utilizada para a coleta e análise dos dados apresentada por [1].

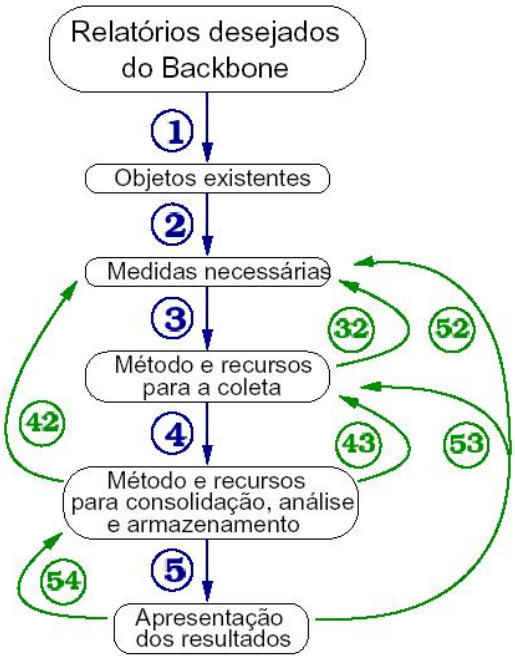

Figura 2.1: Fluxograma

2.1.1 Objetos

Essa é a etapa inicial, a qual consiste na escolha dos objetos do backbone que irão participar direta ou indiretamente dos alvos das medidas. Geralmente estes objetos são os próprio recursos do backbone.

Os objetos podem ser: processamento, memória, tabelas de roteamento, filas, roteadores, interfaces, enlaces, comutadores, modems, multiplexadores, etc.

2.1.2 Variáveis de Medida

Uma vez escolhido o conjunto de objetos e a dependência entre eles, torna-se necessário definir quais variáveis de medida que serão obtidas.

Cada tipo de medida está relacionado a um certo conjunto de objetos. Logo a escolha dos objetos irá definir o universo de medidas possíveis a serem realizadas.

Exemplo: uma medida como tamanho de tabela de roteamento deve ser obtida em objetos que possuam a característica de roteamento. Objetos como enlaces, comutadores, ou modems não podem ser utilizados para tal medida.

2.1.3 Exemplos de Medidas

Comportamento da tabela de roteamento, atraso de um enlace, latência do envio de pacotes entre dois roteadores, caminho de envio de pacotes, utilização de CPU, descarte de pacotes na fila de saída de interface, retransmissão de pacotes, etc.

2.1.4 Método e Recursos para a Coleta

Para a obtenção das medidas escolhidas é necessário definir o método de coleta, que envolva a descrição dos recursos utilizados e o relacionamento destes com os objetos da medida escolhida.

Além dos recursos de hardware e software, o método de coleta deve prover descrições sobre: o tempo de amostragem, o modo de coleta (ativo, passivo, ...), o sincronismo necessário do coletor (ex: NTP, GPS, local, ...), protocolos necessários para obtenção dos dados (ex: SNMP, TELNET, BGP,...) e se for o caso as senhas de acesso.

É necessário redefinir os parâmetros de medida (número 32 da figura) se uma determinada medida descrita no processo número 2 não puder ser realizada por alguma limitação de recursos encontrada no processo número 3.

2.1.5 Análise, Consolidação e armazenamento

Depois de definir o processo de coleta dos dados é necessário então definir qual o tratamento que será feito sobre estes dados e como eles serão armazenados. Este processo será chamado de método de análise, consolidação e armazenamento.

A análise e consolidação dos dados é basicamente um tratamento matemático dos dados coletados, que envolve cálculos tanto no tempo como no espaço. Uma correlação entre medidas diferentes também pode ser feita neste processo. Os dados antes de serem consolidados são chamados de dados "crus".

A definição do método de análise e consolidação deverá conter os cálculos no tempo e no espaço que deverão ser feitos sobre os dados.

Exemplos de consolidação: média do tamanho das filas de um roteador em 1 hora; variância em um dia do atraso (latência) em todos os enlaces ligados aos roteadores de *peering*.

O método deverá também contemplar o tratamento necessário a ser feito nos dados caso ocorra alguma falha temporária no processo de coleta.

O método poderá ainda utilizar artifícios estatísticos para estimar valores coletados.

A descrição dos recursos deverá conter detalhes do hardware e software necessários para o armazenamento, análise e consolidação. Muitas vezes os recursos são compartilhados entre a coleta, análise, consolidação e armazenamento.

Se não houver método de análise e/ou consolidação capazes de obter os relatório desejados, é necessário rever os métodos de coleta e as medidas escolhidas (números 42 e 43 da figura). A revisão dos métodos anteriores pode também ser feita se no futuro for desejado algum resultado adicional ou diferente.

2.1.6 Apresentação dos Resultados

A etapa final é a descrição do método de visualização e apresentação dos dados analisados e consolidados.

Parâmetros podem ser descritos como entrada para melhorar a visualização.

Exemplo: cores, escalas, quantidade e tamanho de gráficos, escolha entre gráficos ou tabelas e escolha do tipo dos gráficos.

Caso a visualização dos resultados não seja satisfatória, devido à velocidade de sua apresentação ou à falta de informações, é possível a revisão dos processos anteriores (números 52, 53 e 54 da figura)

2.2 Fluxos

Os dados do projeto foram obtidos através de fluxos de comunicação. Nesta seção será descrita a característica deste parâmetro de medida.

Um fluxo pode ser definido como um tráfego unidirecional com um conjunto de identificação único de variáveis.

Através da medida do fluxo de comunicação dos pacotes IP é possível obter uma série de informações úteis, tais como:

- Uma vez com o fluxo de dados é possível escolher um melhor ponto de conexão. Por exemplo, no caso da maioria dos pacotes estarem vindo de uma rede ASx, e a conexão existente estar em ASy, é mais adequado trocar o ponto de conexão para ASx.

- Através da quantidade de dados e de pacotes dos fluxos é possível ter uma estimativa da aplicação utilizada.

- Ataques do tipo que se utiliza IPs de origem diferentes do original (IP Spoofing) podem ser detectados através da análise dos fluxos medidos.

- Atualizando as flags do cabeçalho do protocolo TCP (*Transmission Control Protocol*) é possível detectar ataques do tipo DDoS (*Distributed Denied of Services*).

- É possível detectar falhas nos filtros de roteamento.

Os seguintes campos podem ser utilizados para identificar um fluxo: TOS, IP flags, TCP flags, AS origem, AS destino, IP origem, IP destino, Porta origem, Porta destino, Rede IP origem (com a máscara), Rede IP destino (com a máscara), Protocolo de transporte e Protocolo de rede.

Os campos acima descritos podem ser relacionados de forma a definir um fluxo de dados. Inúmeras combinações podem ser feitas. Na figura 2.2 estão os principais tipos de fluxos utilizados em backbones Internet.

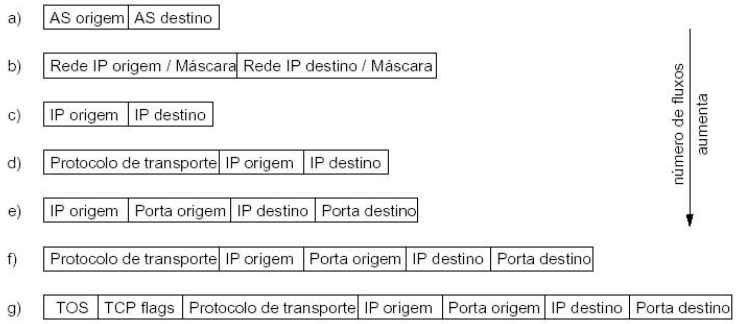

Figura 2.2: Fluxos de comunicação mais comuns

2.2.1 NetFlow

Todos os dados obtidos neste projeto foram exportados pelo roteador de borda da RedeRio localizado no CBPF. Os fluxos são exportados de um roteador Cisco através do NetFlow versão 5. O formato dos pacotes do NetFlow pode ser visto na figura 2.3.

Muitos roteadores e *switches* Cisco suportam serviços NetFlow, que provê uma fonte detalhada de dados sobre tráfego de rede. Os *logs* são utilizados para planejamento da rede, monitoração de performance, cobrança a partir do uso, e muitos outros temas relacionados a segurança, incluindo detecção de intrusão.

Um registro no NetFlow é criado quando o tráfego é visto primeiramente por um roteador Cisco ou *switch* que é configurado para utilizar os serviços do NetFlow. Os fluxos são identificados unicamente pelas características do tráfego que eles representam.

As principais vantagens da utilização do Netflow são as seguintes:

- Sua utilização funciona como *cache* para acelerar os *lookups* nas tabelas de roteamento.
- Com o NetFlow não é necessário verificar as tabelas de *access-list* (apenas de entrada) toda vez que um pacote chega, ficando mais eficiente o processo de roteamento.
- É possível a exportação das informações de fluxo utilizadas pelo *cache* do NetFlow. Isto facilita a coleta de dados para futuras análises sem a necessidade de colocar um analisador em cada enlace.

Para que o NetFlow funcione em um roteador é necessário que a versão do sistema operacional (chamada de IOS) seja compatível [18]. Nos roteadores Cisco não existe um comando geral que habilite o NetFlow para todas as interfaces, logo para sua utilização é necessário a habilitação individual por interface através do comando `route-cache flow`. O processo do NetFlow funciona apenas para os pacotes de entrada, logo os dados exportados pelo NetFlow dizem respeito apenas ao tráfego entrante na interface habilitada.

Para o NetFlow o fluxo é definido como sendo um conjunto de 5 variáveis: O campo `Protocol Type`, IP origem, IP destino, Porta origem e Porta destino. Além destas 5 variáveis as tabelas do NetFlow guardam a interface destino e a interface origem relativas ao trânsito do pacote IP.

A cada pacote IP que entra na interface, o NetFlow identifica o seu fluxo e verifica se já existe uma entrada deste fluxo na tabela de *cache*. Se existir, ele comuta diretamente para a interface destino especificada. Se não existir, ele então realiza um *lookup* nas tabelas de roteamento e nas tabelas de *access-list*. Se este pacote possuir alguma restrição nas tabelas de *access-list* ou se o seu IP destino não for achado nas tabelas de roteamento o pacote então será enviado para a interface `NULL` (um pacote com o destino para interface `NULL` identifica que este foi descartado). O Netflow também cria uma entrada na sua tabela de *cache* para o destino `NULL`.

Outro processo importante no NetFlow é o processo de exportação dos dados, conhecido como *flow-export*. O *flow-export* é feito através do envio de dados encapsulados em pacotes

UDP (*User Datagram Protocol*). Seu destino é o IP do coletor configurado previamente no roteador. O conteúdo do pacote UDP dependerá da versão em que o NetFlow estiver funcionando. Atualmente o roteador pode exportar os fluxos criados pelo NetFlow nas versões de 1 a 8 (A descrição dos pacotes exportados das versões 1, 5, 6 e 8 podem ser vistas em 2.3.2). O momento pelo qual o roteador começa a exportar os dados de fluxo dependerá da configuração. Geralmente o roteador envia uma informação de fluxo assim que ela é expirada. A informação de fluxo se expira assim que sua entrada na tabela do NetFlow é removida. A remoção da tabela do NetFlow pode ocorrer por vários fatores:

- O fluxo criado pelo NetFlow na tabela está ativo por muito tempo (até 30minutos *default*).
- O tamanho da tabela do NetFlow chegou ao limite previamente configurado.
- Não existe mais nenhum tráfego de pacotes no fluxo criado na tabela do Netflow por um certo período de tempo, conhecido como *flow time-out*.
- Conexões TCP que tenham enviado uma mensagem de finalização (FIN) ou tenham enviado uma mensagem de *reset* (RST).

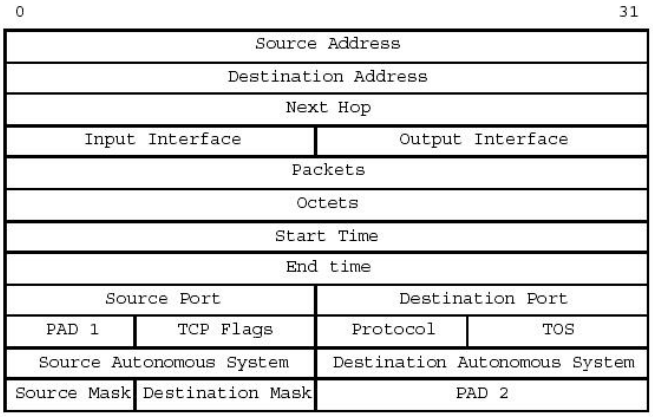

Figura 2.3: Formato dos pacotes Netflow versão 5

2.3 Ferramentas Utilizadas

Nesta seção serão abordadas as principais ferramentas utilizadas neste trabalho para coleta e análise dos dados.

2.3.1 Flow-tools

O Laboratório Office of Information Technology Enterprise Networking Services (OIT/ENS) na Universidade de Ohio (OSU) desenvolveu um conjunto de ferramentas chamado flow-tools [6] para gravar, filtrar e analisar fluxos do Netflow. Essas ferramentas estão listadas abaixo.

Flow-print

Exibe arquivos de flows com dados formatados.

srcIP	dstIP	prot	srcPort	dstPort	octets	packets
131.238.205.199	194.210.13.1	6	6346	40355	221	5
192.5.110.20	128.195.186.5	17	57040	33468	40	1
128.146.1.7	194.85.127.69	17	53	53	64	1
193.170.62.114	132.235.156.242	6	1453	1214	192	4
134.243.5.160	192.129.25.10	6	80	3360	654	7
132.235.156.242	193.170.62.114	6	1214	1453	160	4
130.206.43.51	130.101.99.107	6	3226	80	96	2
206.244.141.3	128.163.62.17	6	35593	80	739	10
206.244.141.3	128.163.62.17	6	35594	80	577	6
212.33.84.160	132.235.152.47	6	1447	1214	192	4

Tabela 2.1: Exemplo do flow-print

Flow-Filter

Filtra flows baseadas em porta, protocolo, AS (*Autonomous System*) number, IP, ToS, TCP bits e tags.

Flow-dscan

Detecção de DoS e scans.

- Detecta hosts que possuem muitas flows para outros hosts.
- Detecta hosts que esteja utilizando grande número de portas TCP/UDP.

srcIP	dstIP	prot	srcPort	dstPort	octets	packets
155.52.46.50	164.107.115.4	6	33225	119	114	2
128.223.220.29	129.137.4.135	6	52745	119	1438382	1022
155.52.46.50	164.107.115.4	6	33225	119	374	6
164.107.115.4	192.58.107.160	6	60141	119	5147961	8876
128.223.220.29	129.137.4.135	6	52745	119	1356325	965
128.223.220.29	129.137.4.135	6	52714	119	561016	398
130.207.244.18	129.22.8.64	6	36033	119	30194	121
155.52.46.50	164.107.115.4	6	33225	119	130	2
198.108.1.146	129.137.4.135	6	17800	119	210720652	216072

Tabela 2.2: Exemplo do flow-filter

- Indicado principalmente para redes menores ou com filtros que não limitem demasiadamente o tráfego.

Flow-Expire

Remove fluxos antigos, baseado na utilização de disco. Utilizado também em ambientes onde o armazenamento é feito em ambiente distribuído.

Flow-stat

- Gera relatório a partir de arquivos de flows.
- Facilidade na importação de dados para programas de geração de gráficos, como o gnuplot.
- Inclui em seus relatórios informações como IP, pares de IPs, portas, número de pacote, número de bytes, next hop, AS, ToS bits, router originador e tags.

Flow-stat-summary

Gera relatórios contendo totais e médias de flows, octetos, pacotes, duração das flows, médida de pacotes das flows, entre outras informações.

2.3.2 Cflowd

O Cflowd é um conjunto de programas e bibliotecas que possibilitam a coleta de dados de fluxo (*flows*) provenientes dos roteadores Cisco que utilizam o Netflow. O conjunto Arts++ foi desenvolvido para auxiliar o Cflowd. A principal função do Arts++ é estabelecer um formato

para o armazenamento de fluxos coletados pelo Cflowd. Além do formato de armazenamento o Arts++ provê também um conjunto de programas de análise. O conjunto de ferramentas do Cflowd e do Arts++ são mantidos pelo CAIDA (*Cooperative Association for Internet Data Analysis*) [5].

O Cflowd é composto basicamente de três programas:

O primeiro programa é responsável pelo tratamento inicial dos dados enviados pelos roteadores chamado de dados *raw*. Este programa é chamado de `cflowdmux`.

O segundo programa que é responsável em manter uma tabela de dados por interface de entrada para cada roteador Cisco, também é responsável para enviar os dados para o coletor central. Este programa é chamado de `cflowd`.

O terceiro programa é responsável por coletar os dados de todas as instâncias do `cflowd` e gravar em disco no formato ARTS. Este programa é chamado de `cfdcollect`.

O programa `cflowdmux`

O `cflowdmux` é responsável pela recepção dos pacotes UDP de *flow-export* enviados pelos roteadores Cisco e o seu armazenamento em uma memória compartilhada (*shared memory*). O `cflowdmux` trabalha com dois *buffers* de pacotes, para que os programas clientes possam ler os dados de um *buffer* enquanto o `cflowdmux` grava os dados em outro *buffer*. Veja a figura 2.4. O `cflowdmux` utiliza semáforos para a alocação dos *buffers*. Assim que os pacotes UDP de *flow-export* chegam, o `cflowdmux` aloca um semáforo para evitar que um processo de leitura esteja lendo um *buffer* de escrita.

O `cflowdmux` não interpreta os dados de fluxo recebidos, logo é necessário que algum programa cliente esteja rodando para interpretar, agrupar e armazenar os dados. A interpretação e agrupamento é feito pelo `cflowd` e o armazenamento pelo `cfdcollect`.

O programa `cflowd`

O programa `cflowd` fica constantemente olhando os semáforos dos *buffers* de pacote criados pelo `cflowdmux`. Se algum semáforo se torna disponível o `cflowd` lê o *buffer* de pacotes, interpretando os dados e os agrupando em tabelas na memória. O processo de agrupamento é

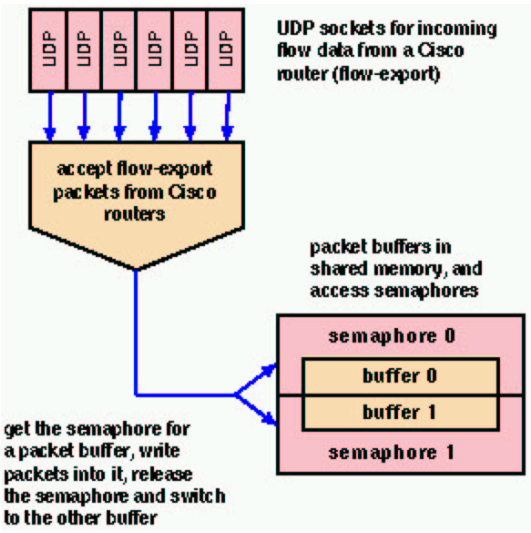

Figura 2.4: O funcionamento básico do `cflowdmux`

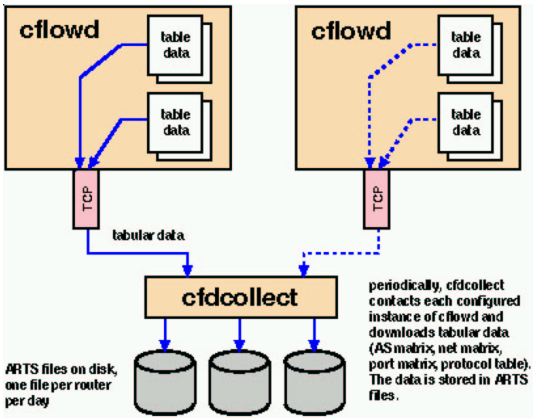

Figura 2.5: O modo de funcionamento do `cfdcollect`

feito através de tabelas na memória e deve ser configurado previamente, quanto maior os tipos de agrupamento maior a quantidade de memória alocada. As tabelas para agrupamento podem ser nos seguintes formatos:

- **asmatrix** : matriz indicando a quantidade de pacotes e bytes por AS origem e AS destino.
- **netmatrix** : matriz indicando a quantidade de pacotes e bytes por rede de origem e rede de destino (a máscara da rede origem e destino é obtida na tabela de roteamento do roteador que exportou os dados de fluxo , logo deve-se tomar cuidado para os backbones que utilizam sumarização).
- **portmatrix** : Bytes e pacotes por porta origem e porta destino.
- **ifmatrix** : Bytes e pacotes por interface origem para interface destino (A interface designada é o número utilizado como `ifIndex` na MIB).
- **protocol** : Tabela mostrando pacotes e bytes por protocolo IP.
- **nexthop** : Tabela mostrando pacotes e bytes por *nexthop*.
- **tos** : Tabela mostrando pacotes e bytes por *Type of Service* (TOS).
- **flows** : Fluxos sem tratamento, ou seja em formato *raw* (consomem muita memória pois não realizam agrupamento).

O tamanho das tabelas com os dados agrupados aumenta a cada vez que o programa `cflowd` lê os dados dos *buffers* e os reconhece como informação pertencente a algum agrupamento. Se por exemplo, o `cflowd` for configurado apenas com o agrupamento **portmatrix** e os dados lidos dos *buffers* representarem apenas pacotes ICMP estes dados não serão agrupados e serão descartados.

Existem duas maneiras de ler os dados agrupados nas tabelas do `cflowd`: localmente ou remotamente. A leitura local é feita através de um socket local e permite que programas rodando localmente acessem as tabelas de agrupamento (como os programas `cfdases` e `cfdnets`). A

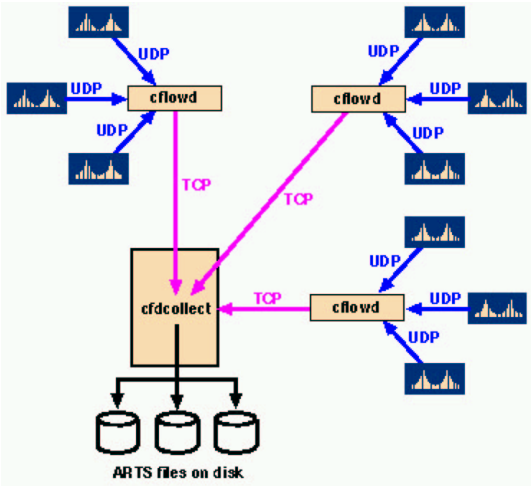

Figura 2.6: A centralização das coletas feita pelo `cfdcollect`

leitura remota é feita através de uma porta TCP previamente especificada (como o programa `cfdcollect`), veja figura 2.5.

Talvez o maior problema da ferramenta esteja no processo de limpeza das tabelas de agrupamento na memória. Esta limpeza só é feita quando existe um programa acessando remotamente o `cflowd`, ou seja com a utilização do `cdfcollect`. Se não existir coleta remota, o programa `cflowd` aloca memória constantemente conforme os novos fluxos vão sendo agrupados. Se não existir novas informações de fluxo a memória não cresce. Porém devido à característica dos ataques DDoS [10] a quantidade de fluxos tem taxa de crescimento muito grande, tornando a tabela na memória infinitamente grande e causando muitas vezes o "crash" do programa `cflowd`.

O programa `cfdcollect`

Este programa é responsável pela conexão TCP até a máquina onde o `cflowd` estiver rodando, e também responsável pela leitura das tabelas de agrupamento e o armazenamento no formato ARTS em disco. Com ele é possível a centralização e o armazenamento das informações coletados individualmente pelos programas `cflowd` localizados remotamente. Veja as figuras 2.5 e 2.6.

Um importante parâmetro a ser configurado no `cfdcollect` é o período de coleta. Este

período de coleta definirá de quanto em quanto tempo o `cfdcollect` irá se conectar ao `cflowd` remoto para pegar as tabelas de agrupamento de fluxo. Este período pode ser crítico quando a taxa de crescimento de fluxos for muito grande e incapaz de ser suportada pela memória da máquina onde o `cflowd` está rodando, isto ocorre durante ataques [10].

Tipos `export-flow` interpretados pelo Cflowd

O Cflowd pode interpretar as versões 1, 5, 6 e 8 do NetFlow. As diferenças entre as versões estão basicamente no formato dos pacotes exportados. O processo de exportação dos pacotes contendo informações de fluxo pelo roteador é praticamente o mesmo, apenas a quantidade de entradas de fluxo é que varia (*flow entry*). Abaixo estão as descrições dos campos de dados dos pacotes UDP exportados pelas versões interpretadas pelo Cflowd. Estas informações foram retiradas do arquivo `CflowdFlowPdu.h`:

- Cabeçalho versão 1:

```
16bits    version        //  flow-export version number
16bits    count          //  number of flow entries
32bits    sysUptime
32bits    unix_secs
32bits    unix_nsecs
```

  Descrição de um fluxo da versão 1 - máximo de 24 entradas de fluxo por cada pacote UDP exportado (máximo 24 *flow entries*):

```
32bits     srcaddr     // source IP address
32bits     dstaddr     // destination IP address
32bits     nexthop     // next hop router's IP address
16bits     input       // input interface index
16bits     output      // output interface index
32bits     pkts        // packets sent in duration
32bits     bytes       // octets sent in duration
32bits     first       // SysUptime at start of flow
32bits     last        // and of last packet of flow
16bits     srcport     // TCP/UDP source port number or equivalent
16bits     dstport     // TCP/UDP destination port number or equivalent
16bits     pad1
8bits      prot        // IP protocol, e.g., 6=TCP, 17=UDP, ...
8bits      tos         // IP Type-of-Service
32bits     pad2
32bits     pad3
```

- Cabeçalho versão 5:

```
16bits    version         //  flow-export version number
16bits    count           //  number of flow entries
32bits    sysUptime
32bits    unix_secs
32bits    unix_nsecs
32bits    flow_sequence   //  sequence number
8bits     engine_type     //  no VIP = 0, VIP2 = 1
8bits     engine_id       //  VIP2 slot number
16bits    reserved        //  unused
```

Descrição de um fluxo da versão 5 - máximo de 30 entradas de fluxo por cada pacote UDP exportado (máximo 30 *flow entries*) :

```
32bits     srcaddr     // source IP address
32bits     dstaddr     // destination IP address
32bits     nexthop     // next hop router's IP address
16bits     input       // input interface index
16bits     output      // output interface index
32bits     pkts        // packets sent in duration
32bits     bytes       // octets sent in duration
32bits     first       // SysUptime at start of flow
32bits     last        // and of last packet of flow
16bits     srcport     // TCP/UDP source port number or equivalent
16bits     dstport     // TCP/UDP destination port number or equivalent
8bits      pad
8bits      tcp_flags   // bitwise OR of all TCP flags in flow; 0x10
                          //  for non-TCP flows
8bits      prot        // IP protocol, e.g., 6=TCP, 17=UDP, ...
8bits      tos         // IP Type-of-Service
16bits     src_as      // originating AS of source address
16bits     dst_as      // originating AS of destination address
8bits      src_mask    // source address prefix mask bits
8bits      dst_mask    // destination address prefix mask bits
16bits     reserved
```

- Cabeçalho versão 6:

```
16bits  version         // version
16bits  count           // the number of records in PDU
32bits  sysUptime       // current time in msecs since router booted
32bits  unix_secs       // current seconds since 0000 UTC 1970
32bits  unix_nsecs      // residual nanoseconds since 0000 UTC 1970
```

```
32bits  flow_sequence   // seq counter of total flows seen
8bits   engine_type     // type of flow switching engine
8bits   engine_id       // ID number of the flow switching engine
16bits  reserved
```

Descrição de um fluxo da versão 6 - máximo de 27 entradas de fluxo por cada pacote UDP exportado (máximo 27 *flow entries*):

```
32bits    srcaddr       // source IP address
32bits    dstaddr       // destination IP address
32bits    nexthop       // next hop router's IP address
16bits    input         // input interface index
16bits    output        // output interface index
32bits    pkts          // packets sent in duration
32bits    bytes         // octets sent in duration
32bits    first         // SysUptime at start of flow
32bits    last          // and of last packet of flow
16bits    srcport       // TCP/UDP source port number or equivalent
16bits    dstport       // TCP/UDP destination port number or equivalent
8bits     rsvd
8bits     tcp_flags     // bitwise OR of all TCP flags seen in flow
8bits     prot          // IP protocol, e.g., 6=TCP, 17=UDP, ...
8bits     tos           // IP Type-of-Service
16bits    src_as        // originating AS of source address
16bits    dst_as        // originating AS of destination address
8bits     src_mask      // source address prefix mask bits
8bits     dst_mask      // destination address prefix mask bits
8bits     in_encaps     // size in bytes of the input encapsulation
8bits     out_encaps    // size in bytes of the output encapsulation
32bits    peer_nexthop  // IP address of the nexthop w/in the peer (FIB)
```

- A versão 8 trabalha com 5 tipos de agregados, abaixo está o descrito o cabeçalho:

```
16bits    version        // flow-export version number
16bits    count          // number of flow entries
32bits    sysUptime      // current time in msecs since router booted
32bits    unix_secs      // current seconds since 0000 UTC 1970
32bits    unix_nsecs     // residual nanoseconds since 0000 UTC 1970
32bits    flow_sequence  // sequence number
8bits     engine_type    // type of flow switching engine
8bits     engine_id      // ID number of the flow switching engine
8bits     agg_method     // aggregation method
8bits     agg_version    // aggregation version
32bits    reserved       // unused
```

Descrição de um fluxo agregado por AS da versão 8 - máximo de 51 entradas de fluxo por cada pacote UDP exportado (máximo 51 *flow entries*) :

```
32bits  flows    // number of flows
32bits  pkts     // number of packets
32bits  bytes    // number of bytes
32bits  first    // sysUptime at start of flow
32bits  last     // sysUptime at end of flow
16bits  src_as   // source AS
16bits  dst_as   // destination AS
16bits  input    // input interface index
16bits  output   // output interface index
```

Descrição de um fluxo agregado por Protocolo e Porta da versão 8 - máximo de 51 entradas de fluxo por cada pacote UDP exportado (máximo 51 *flow entries*):

```
32bits  flows     // number of flows
32bits  pkts      // number of packets
32bits  bytes     // number of bytes
32bits  first     // sysUptime at start of flow
32bits  last      // sysUptime at end of flow
8bits   prot      // IP protocol (TCP=6, UDP=17, etc.)
8bits   pad
16bits  reserved
16bits  srcport   // source port
16bits  dstport   // destination port
```

Descrição de um fluxo agregado por Rede da versão 8 - máximo de 44 entradas de fluxo por cada pacote UDP exportado (máximo 44 *flow entries*):

```
32bits    flows      // number of flows
32bits    pkts       // number of packets
32bits    bytes      // number of bytes
32bits    first      // sysUptime at start of flow
32bits    last       // sysUptime at end of flow
32bits    srcnet     // source network
32bits    dstnet     // destination network
8bits     dst_mask   // destination netmask length (bits)
8bits     src_mask   // source netmask length (bits)
16bits    reserved
16bits    src_as     // source AS
16bits    dst_as     // destination AS
16bits    input      // input interface index
16bits    output     // output interface index
```

Descrição de um fluxo agregado por Rede de origem da versão 8 - máximo de 44 entradas de fluxo por cada pacote UDP exportado (máximo 44 *flow entries*):

```
32bits    flows       // number of flows
32bits    pkts        // number of packets
32bits    bytes       // number of bytes
32bits    first       // sysUptime at start of flow
32bits    last        // sysUptime at end of flow
32bits    srcnet      // source network
8bits     src_mask    // source network mask length (bits)
8bits     pad
16bits    src_as      // source AS
16bits    input       // input interface index
16bits    reserved
```

Descrição de um fluxo agregado por Rede de destino da versão 8 - máximo de 35 entradas de fluxo por cada pacote UDP exportado (máximo 35 *flow entries*):

```
32bits    flows      // number of flows
32bits    pkts       // number of packets
32bits    bytes      // number of bytes
32bits    first      // sysUptime at start of flow
32bits    last       // sysUptime at end of flow
32bits    dst_net    // destination network
8bits     dst_mask   // destination network mask length (bits)
8bits     pad
16bits    dst_as     // destination AS
16bits    output     // output interface index
16bits    reserved
```

2.3.3 Flowscan

O Flowscan é utilizado para armazenar os dados coletados pelo Cflowd e aqueles armazenados na base ARTS no formato RRD (*Round Robin Database*).

O FlowScan consiste de módulos e scripts em Perl que analisam os fluxos de dados exportados. O programa possui as seguintes características:

- mecanismo de coleta de fluxos
- um banco de dados de alta performance
- uma ferramenta de visualização

Ao final da análise, o FlowScan produz imagens que mostram em tempo real o tráfego da rede.

2.3.4 CUFlow

O CUFlow é um módulo do FlowScan. O CUFlow permite que seja feita uma diferenciação de tráfego por protocolos, serviços, TOS, roteador e rede, além de gerar relatórios sobre o tráfego de rede. O programa ainda contém uma ferramenta gráfica chamada CUGrapher, que é uma CGI para a visualização dos dados obtidos pelo FlowScan.

Capítulo 3

Configuração do Coletor

A seguir será apresentado com detalhes como foi feita a instalação e configuração do sistema operacional utilizado e das ferramentas empregadas.

Inicialmente será abordada a instalação do Sistema Operacional, posteriormente algumas ferramentas necessárias para o funcionamento dos coletores e para a apresentação do dados, e finalmente as ferramentas utilizadas para a coleta e análise dos dados.

No processo de instalação utilizado, optou-se por armazenar os arquivos utilizados no diretório /usr/local/down e em seguida descompactá-los no diretório /usr/local/src.

3.1 Equipamentos Utilizados

3.1.1 Microcomputador

Processador	Pentium III DUAL
Clock	1 GHz
Disco Rígido	5 x 18 GB SCSI
Memória RAM	1 GB
Placas de Rede	3 x Placa Ethernet Intel

Tabela 3.1: Configurações de Hardware do Coletor

3.2 Sistema Operacional

Utilizou-se para a instalação do Coletor um CD com o ISO do Linux RedHat 8.0. Foram utilizados cinco discos SCSI de 18G cada um. Dois desses discos estão espelhados utilizando

Ponto de Montagem	Tamanho
/	18
/export	54

Tabela 3.2: Partições utilizadas

software RAID do tipo 1 e os três discos restantes estão utilizando *software* RAID 5. Na Tabela 3.2 estão descritas as partições instaladas:

3.3 Ferramentas Adicionais

Diversos sotfwares foram instalados no coletor para que as ferramentas de coleta e análise pudessem ser utilizadas. O manual de instalação de todos os programas utilizados pode ser encontrado no apêndice G. Abaixo será feita uma pequena descrição desses programas.

3.3.1 ntpdate

O programa ntpdate é utilizado para sincronizar o relógio da máquina coletora. É fundamental que o relógio da máquina esteja correto para que se possa ter certeza que as análise estão sendo feitas nos horários corretos. O sincronismo será realizado a cada 30 minutos.

Este programa pode ser encontrado em:

`ftp://ftp.redhat.com/pub/redhat/linux/8.0/en/os/i386/RedHat/RPMS/ntp-4.1.1a-9.i386.rpm`

Após sua instalação é necessário modificar o crontab da seguinte forma:

$contrab -e

Então adiciona-se na ultima linha do arquivo:

0,30 * * * * /usr/sbin/ntpdate ntp.coppe.ufrj.br >/dev/null

3.3.2 Arts

O conjunto Arts++ é formado por uma série de classes e aplicações escritas em C++ para a manipulação de arquivos gravados no formato ARTS. A estrutura de armazenamento do ARTS foi criada pelo CAIDA para facilitar o armazenamento e a manipulação de dados coletados pelo Cflowd. O Arts++ pode manipular os seguintes tipos de estruturas: AS matrix, net matrix, port matrix, selected port table, protocol table, TOS table, interface matrix, nexthop table,

forward IP path and RTT, BGP4 route table e RTT time series table. Para maiores detalhes veja [7].

Esta biblioteca pode ser encontrado em: `ftp://ftp.caida.org/pub/arts++/`

3.3.3 gcc-2.95.2

O programa Cflowd mantido pelo CAIDA para ser compilado necessita do compilador gcc-2.9X.X.

O compilador gcc pode ser encontrado em: `http://gcc.gnu.org/releases.html`

3.3.4 Módulos do Perl

HTML-TABLE

HTML::Table é utilizado para gerar tabelas HTML para scripts CGI. Utilizando os métodos providos por este módulo, tabelas complexas podem ser criadas, manipuladas e impressas através de scripts em Perl.

Pode ser encontrado em: `http://search.cpan.org/search?dist=html-table`

Digest

Módulo para carregar automaticamente vários módulos do tipo Digest.

Pode ser encontrado em: `http://search.cpan.org/search?dist=digest`

Cflow

Esse módulo em Perl é utilizado pelo FlowScan para ler os arquivos gerados pelo Cflowd ou pelo Flow-tools.

Pode ser encontrado em: `http://net.doit.wisc.edu/ plonka/Cflow/`

Net-Patricia

Esse módulo utiliza a estrutura de dados de Patricia Trie para rapidamente apresentar prefixos de endereços IP encontrados. Utilizado pelo FlowScan.

Pode ser encontrado em: `http://net.doit.wisc.edu/ plonka/Net-Patricia/`

Config_Reader

ConfigReader é um conjunto de classes para a leitura de arquivos de configuração. O programador pode facilmente especificar as diretivas a serem lidas como também seus valores padrão. Esse módulo é utilizado pelo FlowScan.

Pode ser encontrado em: `http://search.cpan.org/search?dist=ConfigReader`

Boulder

Boulder fornece um formato para a transmissão de dados entre um ou mais processos. Utilizado pelo FlowScan.

Pode ser encontrado em: `http://search.cpan.org/search?dist=Boulder`

RRDs

RRDs é um módulo fornecido com o programa RRdTool e também utilizado pelo FlowScan.

3.3.5 rrdtool

RRD é uma abreviatura para *Round Robin Database*. RRD é um sistema para armazenar e mostrar séries no tempo (ex: banda da rede, temperatura do computador, média de consumo da CPU). Ele armazena os dados de forma compacta que não se expande com o tempo e apresenta gráficos úteis processando os dados para forçar o seu condensamento.

Para maiores detalhes veja [9].

Este programa pode ser encontrado em: `http://www.rrdtool.com/download.html`

3.3.6 Apache

O Portal necessita de um servidor web. O servidor escolhido foi o Apache.

Este programa pode ser encontrado em: `http://www.apache.org/dist`

Configuração do Apache

A configuração do Apache diz respeito às modificações que devem ser efetuadas no arquivo /usr/local/apache/conf/httpd.conf.

Informações como nome do servidor web, configurações de segurança e outras devem ser inseridas ou modificadas para o devido funcionamento do servidor.

3.4 Flow-tools

A instalação do Flow-tools foi feita utilizando os binários já compilados. O Flow-tools não necessita de arquivos de configuração. A linha de comando utilizada para a captação e armazenamento foi:

/usr/bin/flow-capture -w /export/data/cflowd/flow-tools/2003/ iplocal/ipremoto/PORTA -S5 -V5 -n 287 -N 0 -R /usr/local/flowscan/bin/export

onde,

-w Indica o diretório que serão armazenados os fluxos recebidos.

-S5 Será anexada no final do nome do arquivo a hora que ele foi gerado, de cinco em cinco minutos.

-V5 Será utilizada a versão 5 do NetFlow.

-n287 Número de arquivos gerados por dia na coleta dos dados.

-N0 Não será feito nenhum tipo de hierarquia no diretório no qual se armazena os fluxos

-R Executa o script export que irá gerar um arquivo no mesmo formato do Cflowd para que o flowscan possa tratar os dados.

Este programa pode ser encontrado em: `http://www.splintered.net/sw/flow-tools/`

O binário para Linux deste programa pode ser encontrado em:
`http://cng.ateneo.net/cng/wyu/software/flow-tools.php`

3.5 Cflowd

Para a instalação do Cflowd foi necessária a instalação do gcc-2.95.2 e das bibliotecas ARTS++. Após sua compilação é necessário editar os arquivos de configuração cflowd.conf e cfdcollect.conf. Nesses arquivos serão configurados: o endereço IP dos roteadores que estão exportando os fluxos, a porta que o deamon cflowd deve escutar, assim como os diretórios no qual os arquivos coletados serão armazenados.

O arquivo de configuração utilizado pode ser encontrado nos apêndices F e E.

Este programa pode ser encontrado em: `ftp://ftp.caida.org/pub/cflowd/`

3.6 Flowscan

Após a instalação do Flowscan deve-se editar o seu arquivo de configuração flowscan.cf.

As principais modificações são:

FlowFileGlob - Diretório no qual os arquivos coletados no formato gerado pelo Cflowd estarão.

ReportClass - Indica qual macro será utilizada para o tratamento dos dados.

WaitSeconds - Quanto tempo o FlowScan aguardará para que o próximo arquivo seja tratado.

O arquivo de configuração utilizado pode ser visto no apêndice B.

Este programa pode ser encontrado em: `http://net.doit.wisc.edu/p̃lonka/FlowScan/`

3.7 CUFlow

O arquivo de configuração do CUFlow é o CUFlow.cf. Este arquivo é localizado no diretório no qual o flowscan foi instalado.

Neste arquivo linhas em branco e palavras escritas após (#) são ignoradas.

As diretivas que podem ser configuradas neste arquivo são:

Router

Por definição, CUFlow não se importa de qual roteador o coletor está recebendo os fluxos. Se for especificado a diretiva *Router* o CUFlow agrega todo o tráfego que ele recebe daquele roteador específico e gera arquivos para o tratamento do RRdTool.

Subnet

Cada entrada *Subnet* no arquivo é composta por um endereço IP/Máscara que representa uma rede local.

Por exemplo a rede da UFRJ é representada por: 146.164.0.0/16.

O CUFlow trata qualquer pacote que seja destinado a uma subnet local como um pacote saindo do roteador. Já os pacotes destinados a redes locais especificadas no arquivo serão tratados como pacotes entrando no roteador.

Network

Cada atribuição *Network* no arquivo é utilizada para gerar arquivos para o RrdTool descrevendo os bytes, pacotes e fluxos entrando e saindo da rede especificada.

Por exemplo a rede da UFRJ: Network 146.164.0.0./16 UFRJ

Service

Cada entrada do tipo *Service* no arquivo deve estar no formato porta/protocolo nome_do_serviço.

Por exemplo para a porta pop: Service 110/tcp pop3

Protocol

Cada entrada do tipo *Protocol* significa que o CUFlow irá gerar estatísticas para cada protocolo especificado.

Por exemplo para TCP: Protocol 6 TCP

TOS

Cada entrada *TOS* significa que o CUFlow irá gerar estatísticas para cada tipo de tráfego com o TOS especificado.

Por exemplo para TOS normal: TOS 0 normal

OutputDir

Esse item especifica o diretório no qual os arquivos de saída serão armazenados.

Por exemplo: OutputDir /export/graficos

Scoreboard

A diretiva *Scoreboard* é utilizada para computar o total de usuários que mais consomem recursos da rede, produzindo um relatório em html.

Por exemplo para computar os 10 usuários que mais consomem recursos: Scoreboard 10 /html/reports /html/current.html

AggregateScore

A diretiva *AggregateScore* indica que o CUFlow deve guardar todas as categorias da diretiva *Scoreboard*, e gerar um relatório geral baseado neste diretiva.

Por exemplo: AggregateScore 10 /html/reports/totals.dat /html/topten.html

O arquivo de configuração utilizado pode ser visto no apêndice C.

Este programa pode ser encontrado em:

`http://www.columbia.edu/acis/networks/advanced/CUFlow/`

Capítulo 4

O Portal

A confecção do portal visa oferecer à comunidade científica e às empresas que utilizam backbones IP, uma fonte de consulta no que diz respeito à caracterização de tráfego, tanto nas suas redes, quanto em outras que possam ser de interesse.

O portal está dividido em diferentes áreas de acordo com os assuntos e a freqüência de atualização. Os seguintes tópicos serão abordados:

Principal - A página principal estará sempre em constante atualização, apresentará notícias recentes, patrocinadores do portal e uma breve explicação sobre o portal.

Equipe - Equipe responsável pela página

Contato - E-mail, telefones e endereços

Ferramentas - Nesta seção são colocadas à disposição diversas ferramentas importantes para a análise de tráfego mas que não foram desenvolvidas no laboratório, serão portanto links dinâmicos que devem ser verificados e atualizados constantemente. As ferramentas estão divididas de acordo com a sua funcionalidade, contendo uma breve descrição de cada ferramenta.

Medidas - Ferramentas para a medição de tráfego.

Utilitários - Utilitários gerais como *tcpdump* e etc.

Visualização - Ferramentas de visualização como *mrtg* e *rrdtool*.

Bibliotecas - Bibliotecas importantes.

Ravel-Lab - Esta seção contém tudo que foi produzido no Laboratório Ravel. A seção está dividida nas três sub-seções abaixo:

Ferramentas - Ferramentas de análise de tráfego desenvolvidas no laboratório

Ravel.

Bibliotecas - Bibliotecas desenvolvidas no laboratório Ravel.

Documentos - Trabalhos e artigos que foram escritos pelo laboratório.

Documentos - Está seção contém documentos, teses e artigos interessantes, além de definições feitas pela comunidade científica sobre análise de tráfego.

Artigos - Artigos interessantes sobre análise de tráfego

Definições - Definições gerais

Links - Está seção mostra diversos links interessantes sobre análise de tráfego com uma breve descrição do portal, necessitando constante atualização.

Projetos - Nesta seção são expostos os projetos que a equipe do site está participando e estará em constante atualização.

Rede-Rio - Descrição do projeto, dados obtidos, estatísticas atuais e etc..

Capítulo 5

Rede Rio

A Rede Rio é uma rede de computadores, integrada por universidades e centros pesquisa localizados no Estado do Rio de Janeiro. Toda parte prática do trabalho foi feita utilizando os dados coletados da RedeRio. Nesta seção será feita uma análise dos dados obtidos.

5.1 Arquitetura do Sistema

A RedeRio é formada por um anel ATM composto por cinco instituições: UFRJ, CBPF, FioCruz, PUC-RIO e Telemar (Ver figura 5.1). Através destas instituições diversas outras universidades e centros de pesquisa tem acesso à rede.

A máquina utilizada para fazer a coleta e análise dos fluxos enviados está localizada no laboratório RAVEL da COPPE/UFRJ (Ver figura 5.2).

Na figura 5.3 é mostrada a rede lógica.

5.2 Análises Interessantes

5.2.1 Segurança

Um aspecto importante no que diz respeito a backbones IP é a segurança da rede sendo um ponto importante o gerenciamento de seus recursos. Quando uma rede tenta prejudicar o desempenho do backbone ou de uma outra rede, ela deve ser reprimida para evitar a queda do desempenho do backbone.

Abaixo serão apresentadas três importantes técnicas de ataque que devem ser monitoradas:

- IP Spoofing: Para se prevenir desse ataque foram desenvolvidas técnicas baseadas no

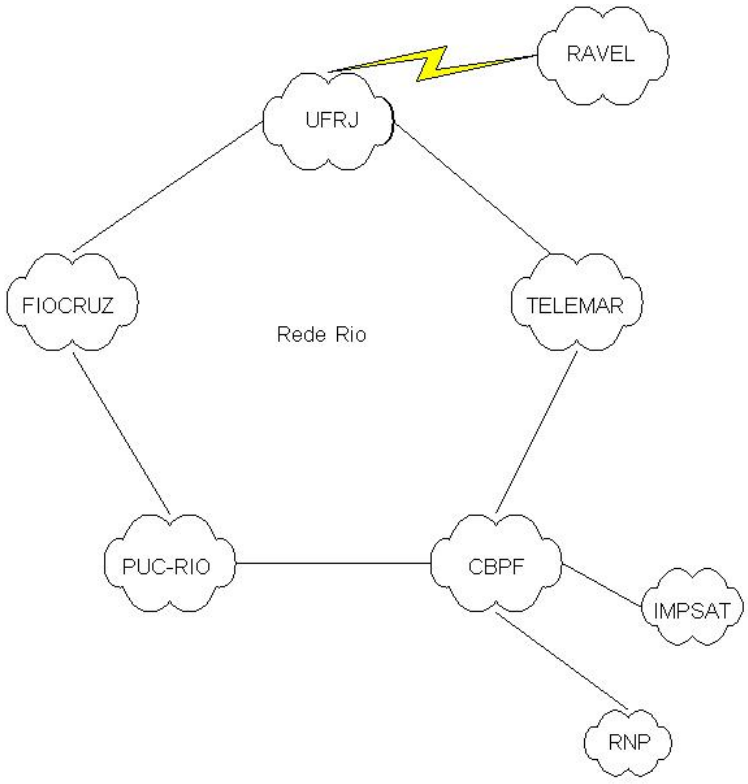

Figura 5.1: Anel ATM da Rede Rio

monitoramento do IP origem dos fluxos. Esta técnica deve ser aplicada as redes origem ligadas aos roteadores de acesso ao backbone.

- IP Flood: O Flood de pacotes é utilizado para degradar ou saturar os recursos de uma rede. Geralmente o atacante tem recursos muito maiores que a vítima, principalmente recursos de enlace e conexões com a Internet. A detecção desse tipo de ataque não é trivial pois depende de vários fatores principalmente das velocidades dos enlaces finais.

- DDoS: Técnica utilizada de forma distribuída. O IP da vítima é único e as redes de origem são diversas, funciona como um IP Flood distribuído utilizando IP Spoofing nas origens. Este ataque geralmente é feito com um pacote TCP utilizando uma flag SYN ativada, simulando um pedido de conexão só que com IP origem falsificado.

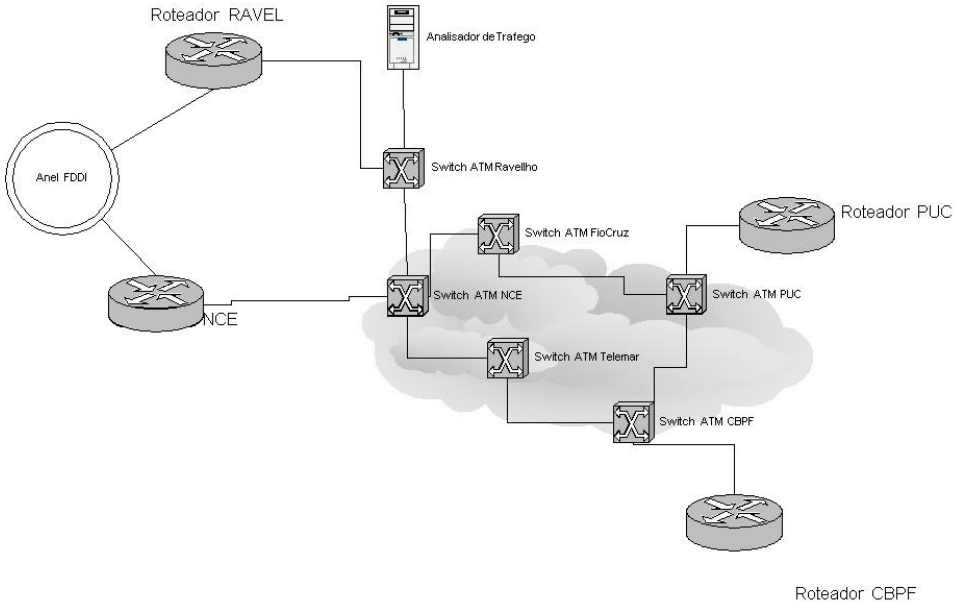

Figura 5.2: Rede Física

Ataques DDoS

Através da análise dos fluxos é possível detectar ataques do tipo DDoS. Fluxos de endereços IPs diferentes para um único IP que crescem muito rápido podem ser considerados como ataque do tipo DDoS. Uma forma de se fazer isso é análisando os fluxos a cada 10 segundos, e se for constatado que nos últimos 10 segundos houve um aumento muito grande de fluxos com a variável IP origem diferente e com a variável IP destino semelhante, pode-se desconfiar de um ataque do tipo DDoS.

Diversos ataques do tipo DDoS foram identificados ao longo deste trabalho. A maioria dos ataques teve origem na UFRJ com destino a endereços IPs localizados na Europa. Tudo indica que o atacante utilizou a rede da UFRJ como "zumbi"para realizar o ataque.

Analisando as figuras 5.4, 5.5 e 5.6 pode-se desconfiar que algum ataque ocorreu. É importante ressaltar que através dos gráficos de bits/s nada podemos dizer em relação a ataques do tipo DDoS. Somente analisando os gráficos de fluxos/s e pacotes/s que podemos perceber o possível ataque.

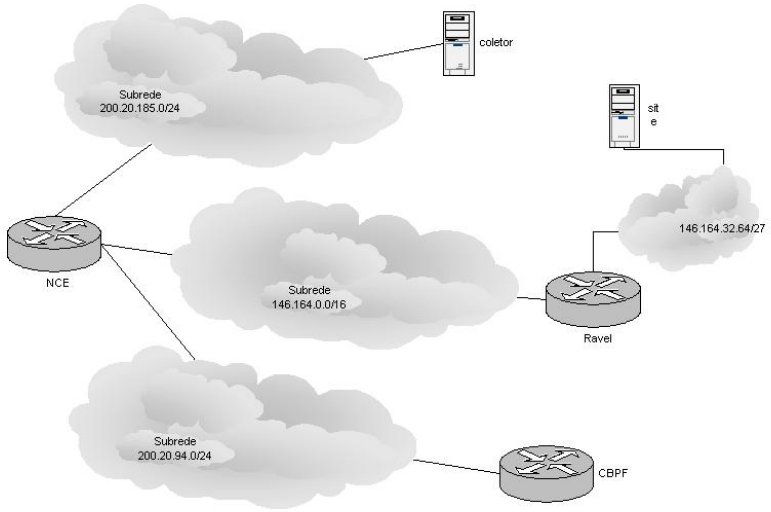

Figura 5.3: Rede Lógica

5.3 Utilização dos Recursos

Nesta seção serão mostrados alguns dados importantes para a caracterização de tráfego obtidos na RedeRio.

Na tabela 5.1 pode ser observado que a grande maioria dos pacotes que trafegam na RedeRio tem tamanho igual ou inferior a 96 bytes.

Na tabela 5.2 observa-se que a grande maioria dos fluxos possuem uma pequena quantidade de pacotes, assim como na tabela 5.2 observa-se que a grande maioria dos fluxos são constituídos de pacotes com tamanho pequeno.

Na tabela 5.4 pode ser observado o tempo de duração dos fluxos. Observa-se que em sua maioria os fluxos tem um tempo de duração pequeno, menor ou igual a 10 segundos.

Na tabela 5.5 observa-se que a web é a aplicação mais utilizada na Rede Rio, seguida pelos softwares de troca de arquivos como Kazaa e E-Donkey.

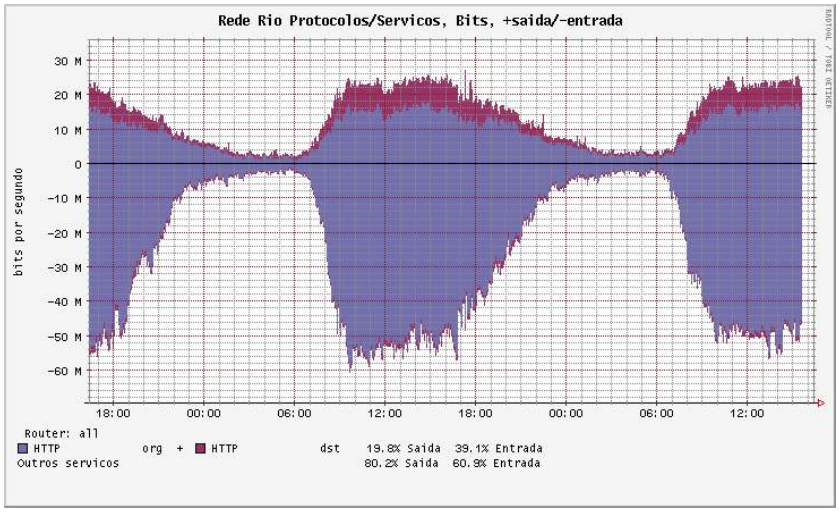

Figura 5.4: Gráfico HTTP de bits/s

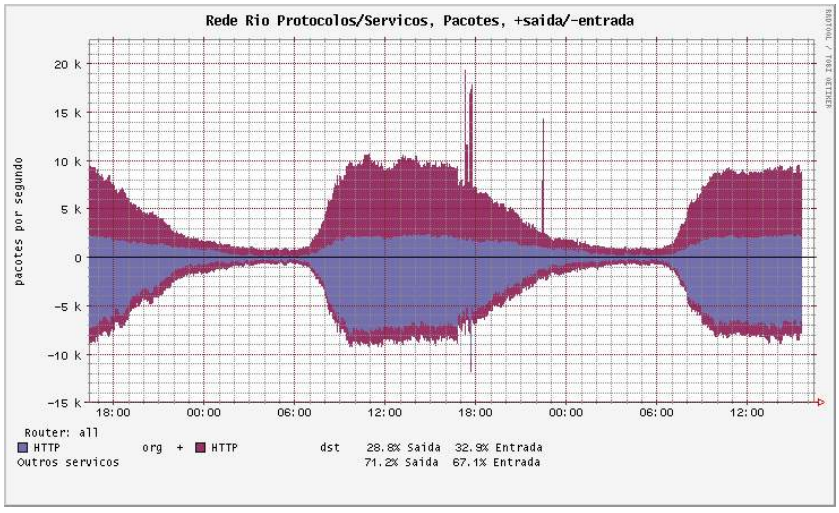

Figura 5.5: Gráfico HTTP de pacotes/s

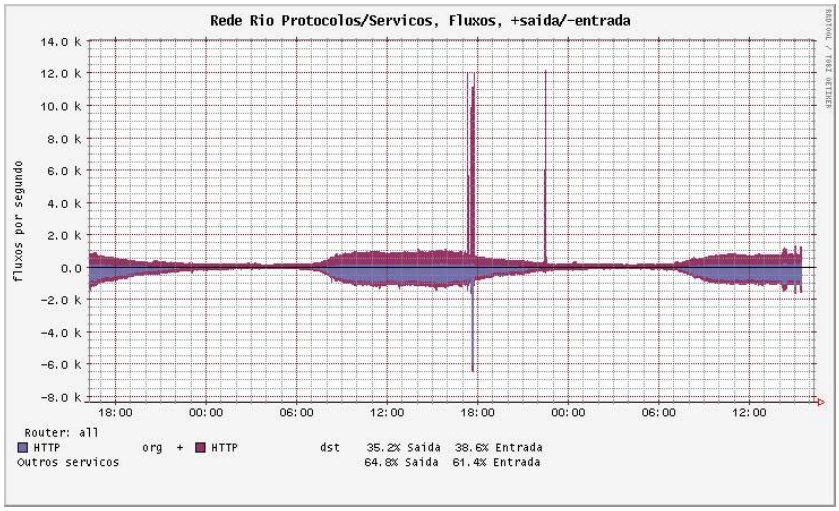

Figura 5.6: Gráfico HTTP de fluxos/s

Tamanho do Pacote	Percentagem
1-32	0.6
64	48.5
96	19.8
128	5.2
160	4.2
192	2.6
224	1.7
256	1.3
288	1.1
320	0.7
352	0.6
384	0.5
416	2.4
448	0.4
480	0.4
512	0.5
544	0.6
576	0.4
1024	3.4
1536	5.2

Tabela 5.1: Distribuição do tamanho do pacote IP

Pacotes	Percentagem
1	45.1
2	16.0
4	15.7
8	11.1
12	3.2
16	1.5
20	1.0
24	0.7
28	0.5
32	0.4
36	0.4
40	0.3
44	0.3
48	0.2
52	0.2
60	0.4
100	1.1
200	1.0
300	0.4
400	0.2
500	0.1
600	0.1
700	0.0
800	0.0
900	0.0
>900	0.1

Tabela 5.2: Distribuição de pacotes por fluxo

Bytes	Percentagem
32	0.3
64	24.9
128	26.5
256	11.8
512	9.7
1280	11.0
2048	3.7
2816	1.6
3584	1.4
4352	0.7
5120	0.8
5888	0.5
6656	0.6
7424	0.4
8192	0.4
8960	0.3
9724	0.3
10496	0.2
11264	0.3
12032	0.2
12800	0.2
13568	0.2
14336	0.2
15104	0.2
15872	0.1
>15872	3.7

Tabela 5.3: Distribuição de bytes por fluxo

Segundos	Percentagem
10	47.1
50	1.7
100	1.3
200	2.2
500	5.8
1000	5.2
2000	4.4
3000	4.2
4000	3.1
5000	1.5
6000	1.6
7000	2.6
8000	0.9
9000	3.4
10000	2.3
12000	1.4
14000	0.8
16000	0.7
18000	0.7
20000	0.6
22000	1.2
24000	0.4
26000	0.3
28000	0.3
30000	0.3
>30000	6.2

Tabela 5.4: Distribuição da duração dos fluxos

Servico	Saída (%)	Entrada (%)
HTTP	18.3	35.1
E-DONKEY	8.9	3.7
KAZAA	8.0	2.9
SMTP	2.2	2.7
FTP	1.7	3.9
IRC	1.5	0.1
HTTPS	0.8	1.0
DNS	0.6	0.9
SSH	0.3	0.7
POP3	0.2	0.3
TELNET	0.1	0.2
REAL	0.0	0.3
OUTROS	57.4	48.2

Tabela 5.5: Percentagem de utilização da banda pelos serviços

Capítulo 6

Conclusão

O ambiente para a coleta de dados foi montado e uma pesquisa sobre a caracterização de tráfego no backbone da RedeRio foi feita, tornando disponível através de um portal as informações coletadas e os resultados obtidos. A intenção é de que o trabalho se estenda e que o portal possa ser uma referência nacional sobre o assunto. O portal está em operação no endereço: `http://iptraf.ravel.ufrj.br`

Modelar o tráfego da Internet é uma questão importante. Só será possível entender as características do tráfego, prever a performance de rede, detectar problemas de segurança, como ataques DDoS, com uma análise de tráfego acurada.

Estudos recentes, veja em [17], mostram que o tráfego da Internet é dominado por pequenos fluxos de tamanho entre 10-20 kB, mas em contraste os fluxos de tamanho muito grande tem sido um dos maiores responsáveis pela utilização de banda. Olhando as tabelas 5.3 e 5.5 pode-se confirmar esse fato. Aproximadamente 60% dos fluxos tem tamanho igual ou menor a 256 bytes e apenas 3.7% dos fluxos tem tamanho igual ou superior a 15872 bytes. Olhando a tabela 5.5 percebe-se que aplicações que em geral possuem fluxos de tamanho grande (ex: ftp, kazaa, E-Donkey) são responsáveis por mais de 20% da banda utilizada, percentagem igual a aplicações que possuem fluxos pequenos (ex: web, ssh, telnet).

Nos últimos anos tem sido observado um grande aumento de ferramentas de compartilhamento de arquivos *peer-to-peer*, como por exemplo: Kazaa, Gnutella, E-Donkey. No trabalho realizado essa tendência foi confirmada, mostrando que quase 20% da tráfego de saída e quase 10% do tráfego de entrada, é gerado por essas aplicações, perdendo apenas para o tráfego web.

Essas aplicações *peer-to-peer* tem modificado significantemente a característica do tráfego, uma vez que os fluxos de dados dessas aplicações em sua maioria contém um grande número de bytes, assim como o tráfego multimídia.

As observações feitas confirmam o que foi analisado em [13], onde a maioria dos fluxos são muito pequenos. Cerca de 50% dos fluxos tem tempo de vida menor ou igual a 10 segundos e cerca de 65% tem duração menor que 15 minutos.

Fica claro que provedores de serviços devem estar cientes do comportamento de fluxos pequenos. Em particular, mecanismos de cache nos roteadores devem estar preparados para lidar com um grande volume de fluxos pequenos. Os provedores também devem ficar cientes que fluxos pequenos contribuem de forma significativa na fração de volume de pacotes e bytes, reduzindo a banda disponível em seus links.

Uma importante análise que pode ser feita futuramente é a relação não só do tamanho dos fluxos como o seu tempo de vida, sendo ambos estudos importantes para compreender o comportamento geral da rede.

6.1 Trabalhos Futuros

Este trabalho representa um início de estudos relacionados a caracterização de tráfego em backbones IP. Diversas medidas que não foram utilizadas neste trabalho podem ser acrescentadas tais como:

- Jitter
- Perda de Pacotes
- Retransmissão
- Vazão
- Tabela de Roteamento
- Maiores estudos sobre segurança
- Analise sobre o tempo de vida dos fluxos

Apêndice A

Modulos do Perl

A instalação dos módulos do perl é mesma para todos eles e se dá da seguinte maneira:

$ gunzip -c <nome_pacote >.tar.gz —tar xf

$ cd <nome_pacote >

$ perl Makefile.PL

$ make

$ make test

$ make install

Todos os módulos requerem a versão 5 do Perl.

Apêndice B

Arquivo de Configuração do FlowScan

flowscan Configuration Directives

FlowFileGlob (REQUIRED) # use this glob (file pattern match) when looking for raw flow files to be

processed, e.g.:

FlowFileGlob /var/local/flows/flows.*:*[0-9] #FlowFileGlob flows.*:*[0-9]

FlowFileGlob /export/data/cflowd/flow-tools/flows.*:*[0-9]

ReportClasses (REQUIRED) # a comma-seperated list of FlowScan report classes, e.g.:

ReportClasses CampusIO

ReportClasses SubNetIO

ReportClasses CampusIO

ReportClasses CUFlow

WaitSeconds (OPTIONAL) # This should be <= the -s"value passed on the command-line to cflowd, e.g.:

WaitSeconds 300

WaitSeconds 30

Verbose (OPTIONAL, non-zero = true) Verbose 1

Apêndice C

Arquivo de Configuração do CUFlow

Subnet 146.164.0.0/16

Subnet 139.82.0.0/16

Subnet 200.222.0.0/16

Subnet 152.84.0.0/16

Subnet 200.20.0.0/16

Subnet 200.156.0.0/16

Subnet 157.86.0.0/16

#UniRio

#Subnet 200.156.25.0/24

#UFF

#Subnet 200.20.0.0/21

Faperj

Subnet 200.6.41.0/24

#RNP

Subnet 200.17.0.0/16

#ANP

Subnet 200.179.25.128/26

#BN

Subnet 200.9.175.0/24

#CFET

Subnet 200.9.149.0/24

Network 146.164.0.0/16 UFRJ

Network 139.82.0.0/16 PUC

Network 152.84.0.0/16 CBPF

Network 200.222.0.0/16 TELEMAR

Network 157.86.0.0/16 FIOCRUZ

Network 200.20.56.0/23,200.222.27/24,200.156.32/19 PRODERJ Network 200.6.41.0/24 FAPERJ

Network 200.20.0.0/21 UFF

Network 200.156.25.0/24 UNI-RIO

OutputDir /export/data/cflowd/flow-tools/graphs

Multicast Scoreboard 10 /export/data/cflowd/flow-tools/reports /usr/local/apache/htdocs/resultados/

AggregateScore 10 /export/data/cflowd/flow-tools/reports/agg.dat /usr/local/apache/htdocs/resultad

#Router 200.20.94.20

Service 20-21/tcp ftp

Service 22/tcp ssh

Service 23/tcp telnet

Service 25/tcp smtp

Service 53/udp,53/tcp dns

Service 80/tcp http

Service 110/tcp pop3

Service 119/tcp nntp

Service 143/tcp imap

Service 412/tcp,412/udp dc

Service 443/tcp https

Service 1214/tcp kazaa

Service 4661-4662/tcp,4665/udp edonkey

Service 5190/tcp aim

Service 6346-6347/tcp gnutella

Service 6665-6669/tcp irc

Service 54320/tcp bo2k

Service 7070/tcp,554/tcp,6970-7170/udp real

Protocol 1 icmp

Protocol 4 ipinip

Protocol 6 tcp

Protocol 17 udp

Protocol 47 gre

Protocol 50 esp Protocol 51 ah Protocol 57 skip Protocol 88 eigrp Protocol 169 Protocol 255

TOS 0 normal TOS 1-255 outros

Apêndice D

Script para exportar o fluxo para o formato ASCII

```
#!/usr/bin/perl
    $file = $ARGV[0];
    if ( $file =  /.*ft-v05\.(\d\d\d\d)-(\d\d)-(\d\d)\.(\d\d)(\d\d)(\d\d)/ ){
    $cflowfile = "flows.".$1.$2.$3."_".$4.":".$5.":".$6;
    $command = "/usr/bin/flow-export -f0 <$file >/export/data/cflowd/flow-tools/$cflowfile";
    print "$command\n"; system($command);
    }
    else {
    print "File $file didn't match\n";
    }
```

Apêndice E

Arquivo de Configuração do cfdcollect

```
#-----------------------------------------------------
    # An example system stanza.
    #-----------------------------------------------------
    system {
    logFacility: local6 # Syslog to local6 facility.
    dataDirectory: /export/data/cflowd/cfdcollect
    filePrefix: arts
    pidFile: /usr/local/arts/etc/cfdcollect.pid
    }
    #-----------------------------------------------------
    # An example cflowd stanza for the case where cflowd is running on the # local host.
    #-----------------------------------------------------
    cflowd {
    host: 200.20.185.2
    tcpCollectPort: 2056
    minPollInterval: 60
    }
    #-----------------------------------------------------
    # An example cflowd stanza for a phony host.
    #-----------------------------------------------------
```

cflowd { # host: cflowd.mydomain.org

tcpCollectPort: 2056 # minPollInterval: 300 # }

Apêndice F

Arquivo de Configuração do cflowd

```
OPTIONS {
    # syslog to local6 facility.
    LOGFACILITY: local6
    TCPCOLLECTPORT: 2056
    PKTBUFSIZE: 2048000
    TABLESOCKFILE: /usr/local/arts/etc/cflowdtable.socket
    FLOWDIR: /export/data/cflowd/flows
    FLOWFILELEN: 2048000
    NUMFLOWFILES: 100
    MINLOGMISSED: 1000
    }
    COLLECTOR {
    HOST: 200.20.185.2 # IP address of central collector
    ADDRESSES: { 200.20.185.2 }
    AUTH: none
    }
    #COLLECTOR {
    # HOST: 198.83.20.25 # IP address of another central collector
    # ADDRESSES: { 198.83.20.25, 204.151.78.12, 204.151.77.25 }
    # AUTH: none }
```

CISCOEXPORTER {

HOST: IP_ROTEADOR

ADDRESSES: { IP_ROTEADOR }

SNMPCOMM: 'public'

CFDATAPORT: Numero_Porta

COLLECT: { protocol, portmatrix, ifmatrix, nexthop, netmatrix, asmatrix, tos, flows }

}

Apêndice G

Manual de Instalação

A organização sugerida para a estrutura dos diretórios é a seguinte:

/usr/local/down - Programas compactados necessários

/usr/local/src - Programas descompactados

/usr/local/PROGRAMA - Local de instalação dos programas

Todos os arquivos de configuração estão detalhados nos apêndices.

Para descompactar todas as ferramentas:

$ cd /usr/local/src

$ tar xfvz ../down/nome_do_programa.tar.gz

A instalação das ferramentas pode ser feita na seguinte ordem:

G.1 Servidor Apache

$ cd /usr/local/src/httpd-2.0.44

$ ./configure –prefix=/usr/local/apache

$ make

$ make install

G.2 perl-5.6.1

$ cd /usr/local/src/perl-5.6.1

$ ./configure –prefix=/usr/local/perl

$ make

$ make install

G.3 RRDTOOL

G.3.1 Bibliotecas

As bibliotecas necessárias para a instalação do RRDTool são:

- GD
- Jpeg-6b
- libpng

Para instalar as bibliotecas:

$ cd /usr/loca/src/nome_da_biblioteca

$ ./configure

$ make

$ make install

G.3.2 Instalação do RRDTool

$ cd /usr/local/src/rrdtool-1.0.40

$ ./configure –enable-shared –prefix=/usr/local/rrdtool

$ make

$ make install site-perl-install

G.4 gcc-2.95.2

Essa versão do gcc é necessária para a instalação do Cflowd.

$ cd /usr/local/src/gcc-2.95.2

$ ./configure –prefix=/usr/local/

$ make

$ make install

G.5 flex-2.5.4

Este programa é necessário para a instalação do Cflowd.

$ cd /usr/loca/src/flex-2.5.4

$ ./configure

$ make

$ make install

G.6 bison-1.35

Este programa é necessário para a instalação do Cflowd.

$ cd /usr/loca/src/bison-1.35

$ ./configure

$ make

$ make install

G.7 ARTS++-1-1-a8

Biblioteca necessária para a instalação do Cflowd.

$ cd /usr/local/src/arts++-1-1-a8

$ ./configure –prefix=/usr/local/arts

$ make

$ make install

G.8 cflowd-2-1-b1

$ cd /usr/local/src/cflowd-2-1-b1

$ ./configure

$ make

$ make install

G.9 FlowScan

G.9.1 Patch para o cflowd

Para o FlowScan funcionar com o cflowd é necessário que se aplique um patch. O pach deve estar localizado no diretorio /usr/local/src.

$ gunzip -c cflowd-2-1-b1.tar.gz —tar xf -

$ cd cflowd-2-1-b1

$ patch -p0 ¡ ../cflowd-2-1-b1-djp.patch

G.9.2 Módulos Perl

Abaixo serão listados os módulos em perl necessários para a instalação do FlowScan.

- RRDs
- Boulder
- ConfigReader::DirectiveStyle
- HTML::Table
- Net::Patricia
- Cflow-1.051

Os módulos listados acima precisam de outros módulos que caso não tenham sido instalados previamente precisarão ser instalados, esses módulos são:

- Crypt-DES
- Digest
- Digest-HMAC
- Digest-SHA

A instalação de todos esse módulos é feita da seguinte maneira:

$ cd /usr/local/src/nome_do_modulo

$ perl Makefile.pl

$ make

$ make test

$ make install

G.9.3 Instalação do FlowScan

Finalmente para instalar o FlowScan:

$ cd /usr/local/src/FlowScan-1.006

$ ./configure –prefix=/usr/local/flowscan

$ make

$ make -n install

$ make install

G.9.4 CUFlow

O CUFlow é um módulo em perl para o FlowScan, desta forma não precisa ser compilado, é necessário apenas copiar seus arquivos para os seguintes diretórios:

$ cd /usr/local/src/CUFlow-1.3

$ cp CUFlow.cf /usr/local/flowscan/bin (diretório de instalação do FlowScan)

$ cp CUFlow.pm /usr/local/flowscan/bin

$ cp CUGrapher.pl /usr/local/apache/cgi-bin/ (diretório de instalação do Apache)

G.10 Flow-Tools

$ cd /usr/local/sr/flow-tools-0.66

$ ./configure –prefix=/usr/local/flow-tools

$ make

$ make install

O Flow-Tools pode ser instalador através do pacote RPM

$ cd /usr/local/down/

$ rpm -i flow-tools-0.66-1.i386.rpm

G.11 Colocando as Ferramentas em Operação

G.11.1 Apache

$ /usr/local/apache/bin/apachectl start

Para visualizar os gráficos: http://IP_do_coletor/cgi-bin/CUGrapher.pl

G.11.2 CFlowd

/usr/local/arts/sbin/cflowdmux

/usr/local/arts/sbin/cflowd

/usr/local/arts/sbin/cfdcollect /usr/local/arts/etc/cfdcollect.conf

G.11.3 Flow-Tools

/usr/bin/flow-tools/flow-capture -w diretorio_onde_os_fluxos estao_armazenados 0/0/PORTA_UDP -S5 -V5 -n 287 -N 0 -R /usr/local/flowscan/bin/export

Obs: Você deve optar pelo Flow-Tools ou pelo Cflowd, os dois não podem funcionar simultaneamente. No projeto em questão optou-se pelo Flow-Tools devido ao sua maior flexibilidade e ao maior número de ferramentas.

G.11.4 FlowScan

/usr/local/flowscan/bin/flowscan CUFlow >>/var/log/flowscan 2>&1 </dev/null & >/dev/null

Referências Bibliográficas

[1] Töpke, Claus Rugani, *Uma Metodologia para Caracterização de Tráfego e Medidas de desempenho em Backbones IP*, tese de mestrado. *Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2001*

[2] Stallings W., *High Speed Networks TCP/IP and ATM design Principles*, Prentice Hall, 1990.

[3] K. Claffy, *Internet Traffic Caracterization*, Ph.D. thesis, University of California, San Diego, 1994.

[4] Christian Huitema, *Routing in the Internet*, Prentice Hall, 1995.

[5] CAIDA, *Cooperative Association for Internet Data Analysis*
http://www.caida.org/

[6] FLOW TOOLS, *http://www.splintered.net/sw/flow-tools/*

[7] Arts++ software - mantido pelo CAIDA, http://www.caida.org/tools/utilities/arts

[8] Cflowd software - mantido pelo CAIDA, http://www.caida.org/tools/utilities/cflowd

[9] RRDTOOL ferramenta para a geraçãod de gráficos,
http://www.rrdtool.com

[10] DDoS, *Distributed Denied Of Services*
http://www.opensourcefirewall.com/ddos_whitepaper_copy.html

[11] Chen, Thomas M. *Internet Performance Monitoring*, Proceedings of the IEEE, Vol. 90, No 9, 2002

[12] Murray, Margaret e K. claffy *Measuring the Immeasurable: Global Internet Measurement Infrastructure*, PAM, 2001

[13] K. Claffy, *Understanding Internet Traffic Streams: Dragonflies and Tortoises*, IEEE Communications Magazine, 2002.

[14] Pereira, Mateus Casanova, *Gerenciamento de Níves de Serviços em Redes TCP/IP Baseado na Linguagem XML*

[15] Moore, David, *Infering Internet DenialofService Activity*, CAIDA

[16] Barakat, Chadi, *A flowbased model for Internet backbone traffic*

[17] Y. Zhang e L. Qiu, *Understanding the End to End Perfomance Impact of RED in a Heterogeneous Environment*, Cornell CS tech, 2000
http://www.aciri.org/floyd/red.html

[18] Cisco White Paper, *NetFlow Services and Aplications*,
http://www.cisco.com/warp/public/cc/cisco/mkt/ios/netflow/tech/napps_wp.pdf