

Medidas e Caracterização de Tráfego de Rede

by
Christian Lyra
Pedro Torres

- [1 Introdução](#)
- [2 Motivação](#)
- [3 Medidas de tráfego](#)
 - [3.1 Usando SNMP](#)
 - [3.2 RRDTOOL e CACTI](#)
- [4 Caracterização de tráfego](#)
 - [4.1 Capturando Tráfego](#)
 - [4.2 Port Accounting com iptables](#)
 - [4.3 Sniffers](#)
 - [4.4 NTop](#)
 - [4.5 NetFLOW](#)
 - [4.6 P2P](#)
 - [4.6.1 L7-Filter](#)
 - [4.6.2 Bandwidth Arbitrator](#)
 - [4.7 IDS](#)
 - [4.7.1 Snort](#)
- [5 Conclusão](#)
- [6 Referências](#)

Introdução

Medir e avaliar são duas ações importantes em praticamente qualquer atividade. Em Redes de Computadores não poderia ser diferente. Veremos, na primeira parte deste documento, como fazer medidas de tráfego de rede, como por exemplo quantidade de bits que passaram por uma determinada interface, e na segunda parte, vamos procurar detalhar e caracterizar esse tráfego, procurando responder perguntas do tipo "do total de tráfego, quanto é tráfego web? quanto é smtp?". Para esse fim utilizaremos ferramentas para o Sistema Operacional GNU/Linux.

A versão atualizada desse documento pode ser encontrada em <http://lyra.soueu.com.br/tiki-index.php?page=MedidaTrafegoRede>.

Motivação

Medir e caracterizar o tráfego de rede são as principais atividades para se implementar e fazer cumprir uma política de uso de um recurso limitado e caro que é o acesso à internet. Com o advento das aplicações Peer-to-Peer (P2P), worms e outros tipos de ataques DOS, caracterizar o trânsito passou a ser não só uma questão de planejar e dimensionar, mas sim de prevenir abusos e aumentar a segurança.

Medidas de Tráfego

Tráfego pode ser definido como a quantidade de dados que atravessam um circuito em um certo período de tempo. Usualmente medimos o tráfego em bits (ou megabits) por segundo. Essa informação é essencial para o administrador de redes, pois dá uma visão geral de como a rede está se comportando. Veremos a seguir como medir e criar gráficos de utilização de banda.

Usando SNMP

Nessa seção faremos uma breve introdução ao SNMP e como obter dados importantes de dispositivos com suporte a SNMP.

SNMP

O SNMP, Simple Network Management Protocol, ou Protocolo Simples de Gerenciamento de Rede, é um protocolo que permite a gerência de dispositivos através da rede. O "simples" do nome se refere ao fato do protocolo ser leve e fácil de ser implementando nos mais diversos dispositivos.

Um dispositivo que responde a requisições SNMP é chamado de Agente SNMP, e as informações que ele é capaz de prover são organizadas no que chamamos de MIBs (Management Information Base). Uma tabela MIB define "índices", chamados de OIDs, e conteúdos. Exemplo: o OID `.1.3.6.1.2.1.1.4.0` contém como valor uma string com o contato técnico responsável pelo agente SNMP. Os OIDs são organizados em forma de árvore, e cada ramo da árvore pode receber um nome. O OID do exemplo acima também é conhecido por `SNMPv2-MIB::sysContact.0`, que por sua vez é uma abreviação de `.iso.org.dod.internet.mgmt.mib-2.system.SNMPv2-MIB.sysContact.0`. A maioria dos dispositivos de rede com suporte a SNMP implementa pelo menos a `SNMPv2-MIB`, que contém entre outras coisas a descrição das interfaces de rede e o valor dos contadores dessa interface, permitindo assim que ferramentas `snmp` consultem esses valores para gerar estatísticas de tráfego.

O protocolo SNMP foi ao longo do tempo sendo modificado o que levou a criação de 3 principais versões. A versão 1, a versão 2c, que é a mais usual, e a versão 3. Utilizaremos aqui, sempre que possível a versão 2c, por ser a mais popular.

Segurança e SNMP

O protocolo SNMP foi criado para ser simples e não seguro. Ele usa como protocolo de transporte o UDP, e utiliza como uma espécie de senha as chamadas communities. Uma community nada mais é do que uma string utilizada pelo agente SNMP para identificar a que tipo de dados um gerente SNMP vai ter acesso. A versão 3 do protocolo SNMP implementa um esquema de segurança mais robusto e com suporte a usuário e senha. Justamente por não ser muito seguro, recomendamos que nos roteadores com agente SNMP se restrinja os IPs ao qual o roteador responde requisições SNMP. Além é claro de não se utilizar a community padrão que normalmente vem na configuração padrão de muitos dispositivos, a community "public".

Fazendo consultas SNMP

Veremos a seguir como fazer consultas SNMP utilizando as ferramentas do pacote net-snmp. No Debian esse pacote pode ser instalado com um simples:

```
# apt-get install snmp
```

O pacote net-snmp (antigo ucd-snmp) contém um conjunto de executáveis para a linha de comando para se consultar e configurar agentes SNMP. Desse conjunto veremos apenas dois deles, o snmpwalk e o snmpget, pois são os que nos interessam para coletar informações de um agente SNMP.

snmpwalk

O comando snmpwalk é utilizado para se recuperar uma "árvore" de informações de um agente SNMP. Para tanto ele envia uma série de requisições "SNMP GETNEXT" ao agente. Caso nenhum OID (ou ramo) seja passado na linha de comando, o snmpwalk caminhará pela MIB-2. A utilização do comando é bastante simples, e em geral basta informar a community (com a opção -c), a versão do protocolo SNMP (com a opção -v) e o agente SNMP (ou host). Exemplo:

```
# snmpwalk -c public -v 2c localhost
```

Esse comando irá retornar uma quantidade bem grande de informações, um trecho de um exemplo de saída é mostrado logo abaixo:

```
IF-MIB::ifNumber.0 = INTEGER: 4  
IF-MIB::ifIndex.1 = INTEGER: 1  
IF-MIB::ifIndex.2 = INTEGER: 2
```

```

IF-MIB::ifDescr.1 = STRING: lo
IF-MIB::ifDescr.2 = STRING: eth0
IF-MIB::ifType.1 = INTEGER: softwareLoopback(24)
IF-MIB::ifType.2 = INTEGER: ethernetCsmacd(6)
IF-MIB::ifMtu.1 = INTEGER: 16436
IF-MIB::ifMtu.2 = INTEGER: 1500
IF-MIB::ifSpeed.1 = Gauge32: 10000000
IF-MIB::ifSpeed.2 = Gauge32: 100000000
IF-MIB::ifPhysAddress.1 = STRING:
IF-MIB::ifPhysAddress.2 = STRING: 0:2:55:5d:7:c6
IF-MIB::ifAdminStatus.1 = INTEGER: up(1)
IF-MIB::ifAdminStatus.2 = INTEGER: up(1)
IF-MIB::ifOperStatus.1 = INTEGER: up(1)
IF-MIB::ifOperStatus.2 = INTEGER: up(1)
IF-MIB::ifInOctets.1 = Counter32: 300407800
IF-MIB::ifInOctets.2 = Counter32: 240594264
IF-MIB::ifInUcastPkts.1 = Counter32: 163893
IF-MIB::ifInUcastPkts.2 = Counter32: 396747
IF-MIB::ifInDiscards.1 = Counter32: 0
IF-MIB::ifInDiscards.2 = Counter32: 0
IF-MIB::ifInErrors.1 = Counter32: 0
IF-MIB::ifInErrors.2 = Counter32: 0
IF-MIB::ifOutOctets.1 = Counter32: 300410163
IF-MIB::ifOutOctets.2 = Counter32: 40068731
IF-MIB::ifOutUcastPkts.1 = Counter32: 163925
IF-MIB::ifOutUcastPkts.2 = Counter32: 227667
IF-MIB::ifOutDiscards.1 = Counter32: 0
IF-MIB::ifOutDiscards.2 = Counter32: 0
IF-MIB::ifOutErrors.1 = Counter32: 0
IF-MIB::ifOutErrors.2 = Counter32: 0

```

No trecho acima vemos informações relativas a duas interfaces de rede, como seus estados, a quantidade de octetos que entraram e saíram e o número de erros.

snmpget

O comando `snmpget` é bastante parecido com o `snmpwalk`, mas ao invés de retornar uma árvore inteira, ele consulta apenas um OID. Exemplo:

```

# snmpget -v 2c -c public localhost IF-MIB::ifOutOctets.1
IF-MIB::ifOutOctets.1 = Counter32: 303312917

```

Podemos especificar o OID utilizando nomes ou números.

Veremos na próxima seção ferramentas mais completas capazes de fazer consultas SNMP e gerar gráficos.

RRDTOOL e CACTI

Veremos a seguir como instalar e utilizar duas ferramentas que permitem a aquisição de dados e plotagem de gráficos de informações obtidas através de consultas SNMP, scripts, etc.

RRDTOOL

RRD é a sigla para Round Robin Database. O RRD é um sistema para armazenar e mostrar dados em série obtidos em um determinado período de tempo (banda de rede, temperatura da máquina, etc). Os dados são armazenados de maneira bastante compacta e não aumentam com o tempo (por isso que o banco é dito "circular"). O RRDTOOL também é capaz de gerar gráficos a partir desses dados. Como o RRDTOOL não é capaz de fazer o "polling" dos dados, nem apresentá-los de maneira automática, é bastante comum a sua utilização associada a um front-end.

Como iremos utilizar o RRDTOOL com o front-end CACTI não abordaremos aqui detalhes de sua utilização.

Obtendo e Instalando o RRDTOOL

O RRDTOOL pode ser obtido a partir do site <http://www.rrdtool.com>. No caso do Debian ele pode ser instalado com o comando:

```
# apt-get install rrdtool
```

CACTI

O Cacti é um front-end para o RRDTOOL escrito em PHP. Ele possui uma interface web e armazena seus dados em uma base MySQL. Através do Cacti podemos fazer o polling de hosts SNMP, criar gráficos e gerenciar o acesso de usuários à esses gráficos. O Cacti pode substituir com vantagens a ferramenta MRTG popularmente utilizada para se criar gráficos de utilização de banda. Veremos a seguir como instalar e configurar o Cacti.

Pré-requisitos

Antes de instalar o Cacti é necessário que a máquina já possua instalado e configurado os seguintes pacotes:

- Apache (ou outro servidor web)
- PHP (versão > 4) + extensões php-snmp e php-gd2
- Banco de dados MySQL?
- net-snmp
- rrdtool

Obtendo e Instalando o Cacti

O Cacti pode ser obtido do site <http://www.raxnet.net/products/cacti/>. A versão mais recente quando esse documento foi escrito era a 0.8.3a. Uma vez feito o download do cacti, descompacte-o no local de instalação (sugestão: /usr/local/cacti), com o comando:

```
# tar xvfz cacti-x.x.x.tar.gz
```

Em seguida crie o banco de dados cacti no mysql:

```
# mysqladmin --user=root create cacti
```

E importe a base padrão de dados:

```
# mysql -p cacti < cacti.sql
```

Crie e dê permissões de acesso ao banco cacti ao usuario cactiuser:

```
# mysql -p
GRANT ALL ON cacti.* TO cactiuser@localhost IDENTIFIED BY 'somepassword';
flush privileges;
```

E altere o arquivo de configuração include/config.php de acordo:

```
$database_default = "cacti";
$database_hostname = "localhost";
$database_username = "cactiuser";
$database_password = "somepassword";
```

Crie um usuário no sistema para ser utilizado pelo cacti:

```
# adduser cacti
```

(ou `adduser --home /usr/local/cacti cacti`)

Esse usuário será utilizado para criar o arquivos rra (do rrdtool) e fazer o polling dos hosts snmp. Dê permissão para esse usuário nos diretórios rra e log:

```
# chown -R cacti:cacti rra/ log/
```

E acrescente a seguinte linha ao `/etc/crontab`

```
*/5 * * * * cacti php4 /path_to_cacti/cmd.php > /dev/null 2>&1
```

Finalmente modifique o seu apache para acessar esse diretório (para o nosso exemplo poderíamos acrescentar o seguinte ao `httpd.conf`):

Alias `/cacti/ /usr/local/cacti/`

```
<Directory /usr/local/cacti >
  Options Indexes MultiViews
  AllowOverride None
  Order allow,deny
  Allow from all
</Directory >
```

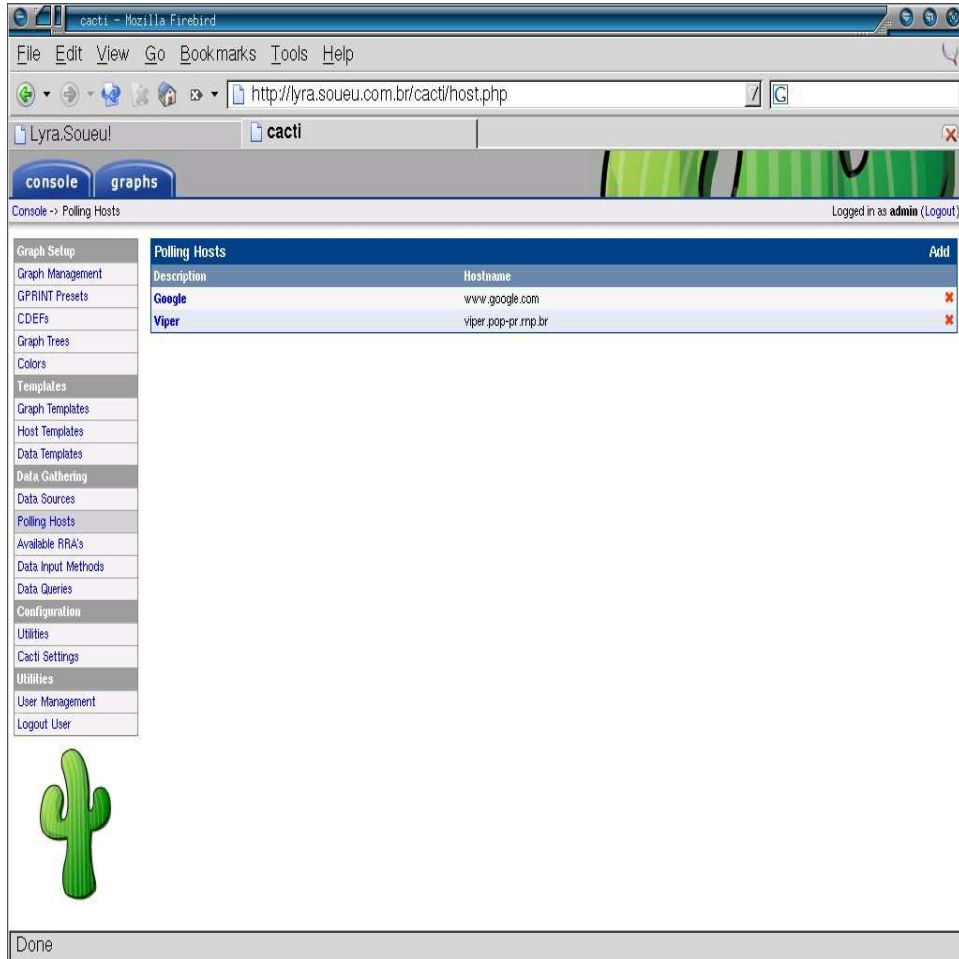
Configurando o Cacti

A configuração do Cacti é bastante simples e é feita via interface web. Veremos a seguir alguns ajustes a serem feitos na configuração padrão e como criar um gráfico de utilização de banda.

Após a instalação é importante acertar os "paths" do Cacti. Para acertar esses parâmetros clique no link "Cacti Settings" do menu "Configuration" e em seguida no link para a aba "Path". Seguindo o nosso exemplo, poderíamos deixar assim:

cacti Web Root	/cacti
Web Server Document Root	/usr/local
snmpwalk Binary Path	/usr/bin/snmpwalk
snmpget Binary Path	/usr/bin/snmpget
rrdtool Binary Path	/usr/bin/rrdtool
PHP Binary Path	/usr/bin/php4

Para criar um gráfico, primeiro é necessário adicionar/selecionar um host (agente SNMP) ao Cacti. Podemos fazer isso clicando no link "polling host" do menu "Data Gathering". Em seguida selecionamos um host ou clicamos no link "Add" para adicionar um novo host. Obs: Clique na imagem para ampliar.




console graphs

Console -> Polling Hosts Logged in as admin (Logout)

Polling Hosts		Add
Description	Hostname	
Google	www.google.com	x
Viper	viper.ppp-pr.rnp.br	x

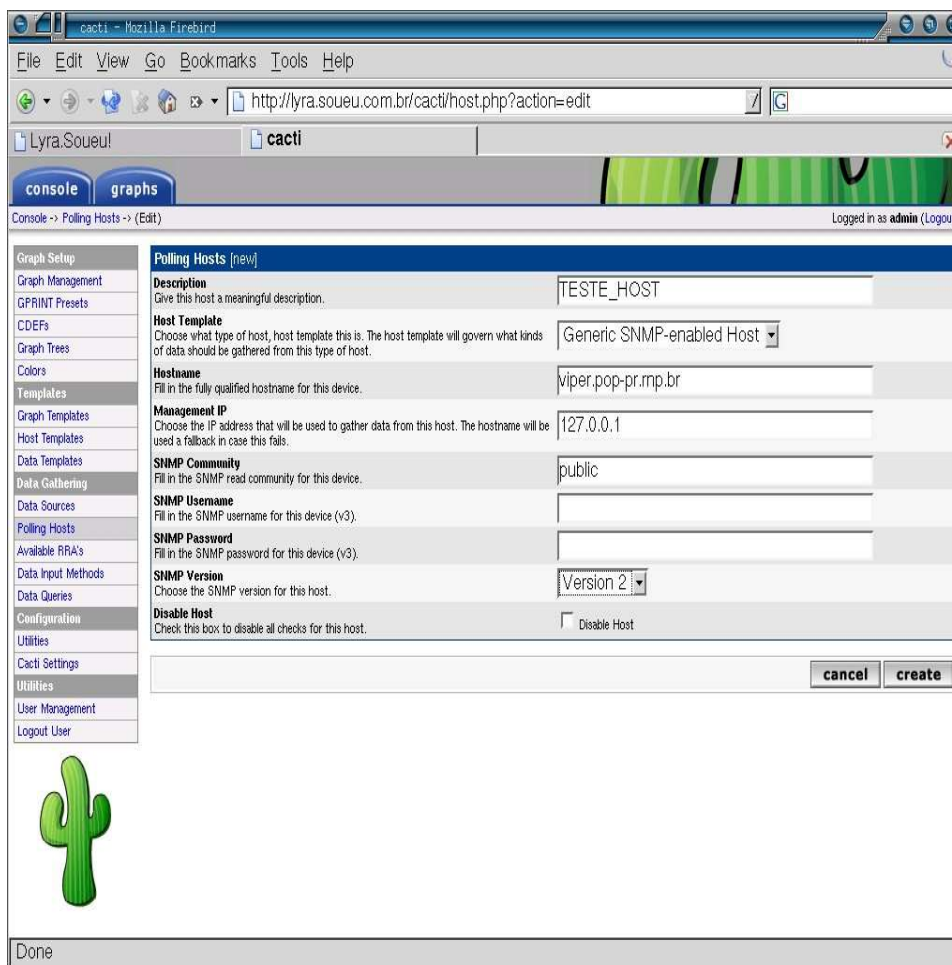
Graph Setup

- Graph Management
- GPRINT Presets
- CDEFs
- Graph Trees
- Colors
- Templates
 - Graph Templates
 - Host Templates
 - Data Templates
 - Data Gathering
 - Data Sources
 - Polling Hosts**
 - Available RRA's
 - Data Input Methods
 - Data Queries
- Configuration
 - Utilities
 - Cacti Settings
- Utilities
 - User Management
 - Logout User



Done

Após clicarmos no "Add" devemos preencher os dados do host em questão, como no exemplo abaixo:

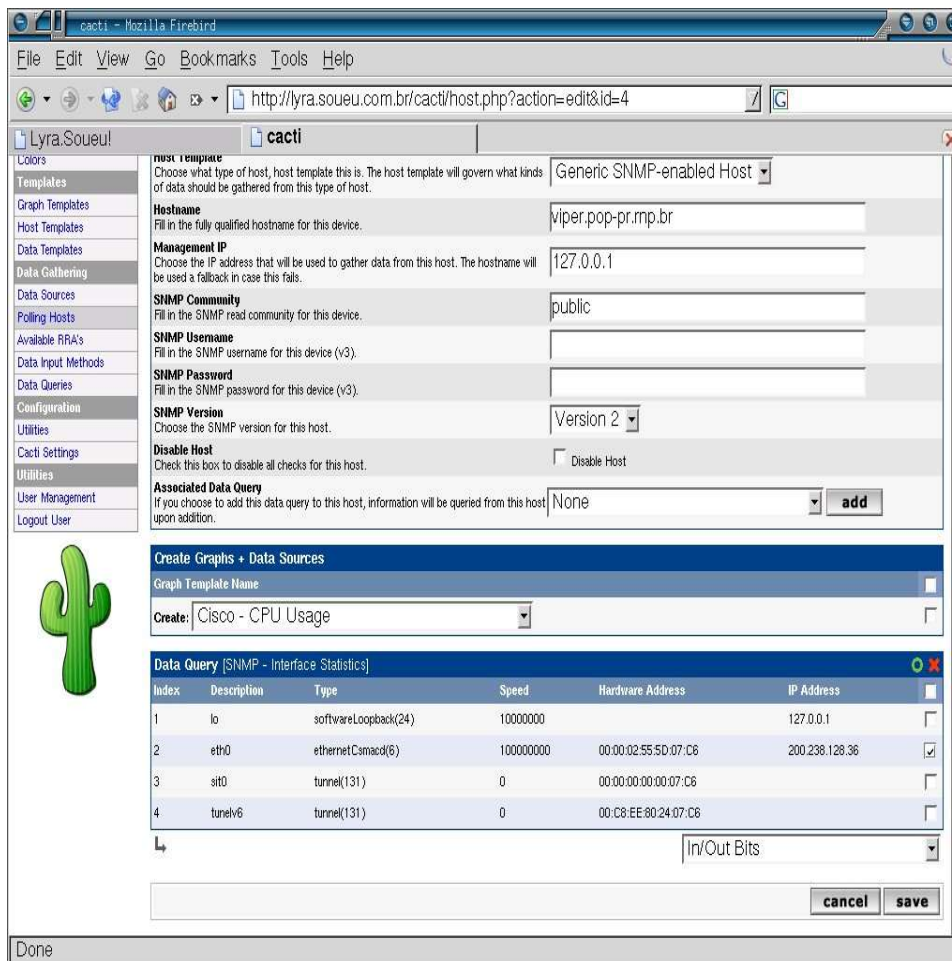


The screenshot shows the Cacti web interface in a Mozilla Firefox browser. The address bar displays the URL: `http://lyra.soueu.com.br/cacti/host.php?action=edit`. The page title is "cacti". The interface includes a navigation menu with "console" and "graphs" tabs. The main content area is titled "Polling Hosts [new]" and contains the following configuration fields:

- Description:** Give this host a meaningful description. Value: `TESTE_HOST`
- Host Template:** Choose what type of host, host template this is. The host template will govern what kinds of data should be gathered from this type of host. Value: `Generic SNMP-enabled Host`
- Hostname:** Fill in the fully qualified hostname for this device. Value: `viper.pop-pr.mp.br`
- Management IP:** Choose the IP address that will be used to gather data from this host. The hostname will be used a fallback in case this fails. Value: `127.0.0.1`
- SNMP Community:** Fill in the SNMP read community for this device. Value: `public`
- SNMP Username:** Fill in the SNMP username for this device (v3). Value: (empty)
- SNMP Password:** Fill in the SNMP password for this device (v3). Value: (empty)
- SNMP Version:** Choose the SNMP version for this host. Value: `Version 2`
- Disable Host:** Check this box to disable all checks for this host. Disable Host

At the bottom of the form are "cancel" and "create" buttons. A small cactus icon is visible in the bottom left corner of the page. The status bar at the bottom of the browser window shows "Done".

Criamos esse host clicando no botão "Create". O Cacti nos mostrará a mesma página novamente, mas acrescentará as interfaces que ele descobriu ao fazer um "snmpwalk" no host. Selecionamos então a interface para a qual queremos criar o gráfico e clicamos em "Add".



The screenshot shows the Cacti web interface in a Mozilla Firefox browser window. The address bar shows the URL: `http://lyra.soueu.com.br/cacti/host.php?action=edit&id=4`. The page title is "Lyra.Soueu! cacti".

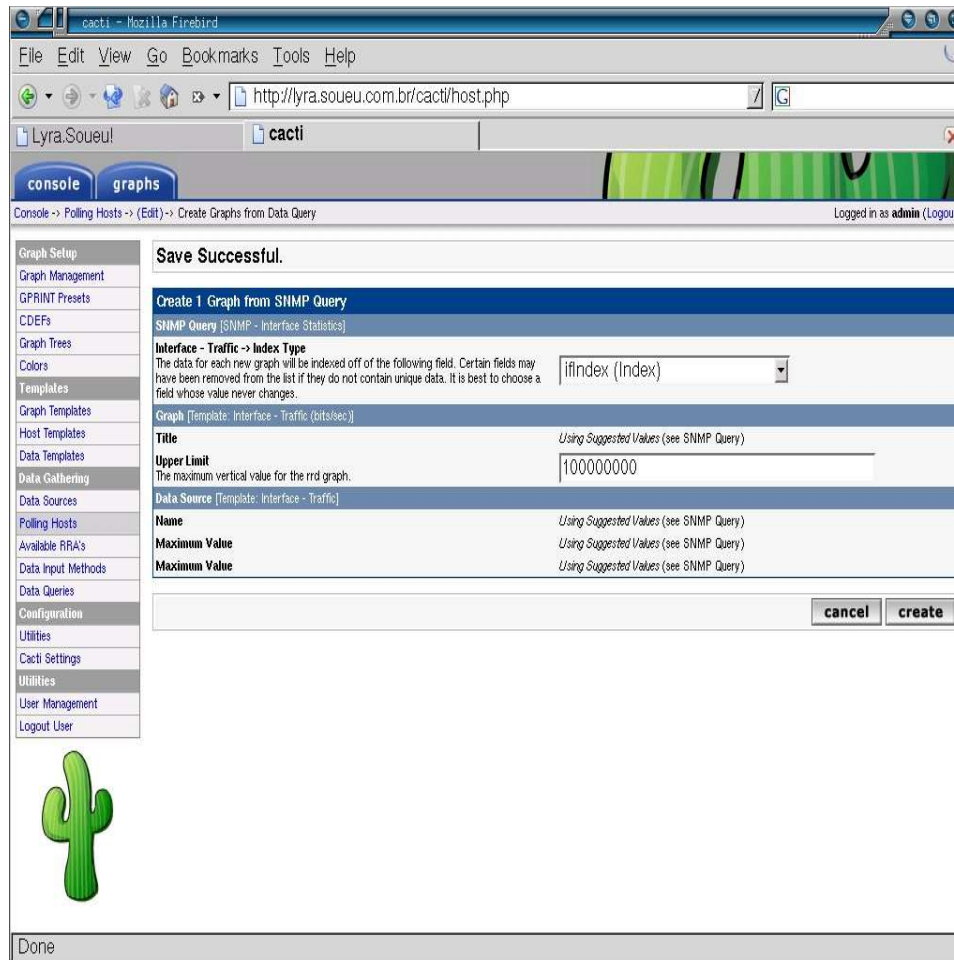
The main content area is divided into two sections:

- Host Configuration:** This section allows editing host details. The "Host Template" is set to "Generic SNMP-enabled Host". Other fields include:
 - Hostname:** viper.pop-pr.mp.br
 - Management IP:** 127.0.0.1
 - SNMP Community:** public
 - SNMP Username:** (empty)
 - SNMP Password:** (empty)
 - SNMP Version:** Version 2
 - Disable Host:** Disable Host
 - Associated Data Query:** None
- Create Graphs + Data Sources:** This section is used to create new data sources for the host. The "Graph Template Name" is set to "Cisco - CPU Usage". Below this is a table titled "Data Query [SNMP - Interface Statistics]" showing discovered interfaces:

Index	Description	Type	Speed	Hardware Address	IP Address	
1	lo	softwareLoopback(24)	10000000		127.0.0.1	<input type="checkbox"/>
2	eth0	ethernetCsmacd(6)	100000000	00:00:02:55:5D:07:C6	200.238.128.36	<input checked="" type="checkbox"/>
3	sit0	tunnel(131)	0	00:00:00:00:00:07:C6		<input type="checkbox"/>
4	tunel6	tunnel(131)	0	00:C8:EE:80:24:07:C6		<input type="checkbox"/>

Below the table, there is a dropdown menu set to "In/Out Bits" and "cancel" and "save" buttons.

Preenchemos os dados referentes ao gráfico que estamos criando:



E pronto! O gráfico está criado. Para ver o resultado clique na aba "Graphs" e em seguida no gráfico que você deseja ver.

O Cacti possui uma série de recursos que não abordaremos aqui, entre eles a possibilidade de criar usuários e selecionar quais gráficos esses usuários podem ver, organizar os gráficos em "árvores", utilizar scripts personalizados para recuperar dados de hosts, etc. Deixaremos para o leitor explorar o Cacti :-).

Caracterização de Tráfego

No capítulo anterior vimos como quantificar o tráfego de rede. Neste capítulo vamos "olhar" o tráfego mais de perto e vamos procurar obter mais detalhes desse tráfego. Nos interessa agora uma análise mais qualitativa do tráfego, ou seja, que tipos de fluxos temos no nosso tráfego? WWW, e-mail, ftp, P2P? Essa análise é importante porque ela vai nos permitir verificar se uma determinada política de uso está sendo cumprida ou não. Vai nos permitir também reforçar a segurança, detectar abusos, ataques de negação de serviço, etc.

Capturando Tráfego

Para analisar o tráfego de rede é necessário que de alguma forma tenhamos acesso a esse tráfego. Veremos a seguir algumas formas de se conseguir isso.

Switch Mirror

Também conhecido como "port mirroring", é um recurso encontrado em alguns switches que permite com que todo o tráfego que passa por uma determinada porta, seja "espelhado" em outra porta.

Vantagens: não interfere no tráfego.

Desvantagens: como uma porta full-duplex 100Mbps pode suportar até 200Mbps de tráfego, pode haver perda de dados na porta espelhada. Alguns switches tem a performance degradada quando se ativa esse recurso.

HUB

Podemos também inserir um HUB ethernet entre segmentos de rede afim de capturar o tráfego.

Vantagens: Simples, barato, não requer modificação de configurações, nem tem impacto sobre a performance dos switches como no caso anterior.

Desvantagens: funciona half-duplex, cria-se um ponto único de falha, e pode levar a colisões quando o tráfego ultrapassar 50% da capacidade máxima.

Network Taps

Network Taps são dispositivos especialmente criados para se capturar/espelhar tráfego. Sua utilização é similar à do HUB, ou seja, devem ser colocados entre dois segmentos de rede, mas ao contrário do HUB eles são Full-Duplex e o tráfego é espelhado em duas portas (RX e TX).

Vantagens: não exigem configuração especial e não impactam a performance da rede. Funcionam de maneira "stealth" (nada é mudado).

Desvantagens: O tráfego é dividido em RX e TX, o que exige que o analisador de tráfego possua duas placas de rede e seja capaz de "re-combinar" o tráfego. Representam também um custo adicional.

Bridges/Roteadores *nix

Uma máquina *nix pode ser configurada como um roteador, ou como uma bridge, e pode capturar/analisar o tráfego que passa por ela.

Vantagens: não exige configuração especial (no caso de se usar uma bridge). Método fácil e barato.

Desvantagens: cria-se um ponto único de falha. A performance da máquina pode interferir na performance da rede.

Netflow

O Netflow é um recurso presente em alguns roteadores que permite que o próprio roteador analise o tráfego e exporte via mensagens Netflow dados sobre os fluxos que atravessam o roteador.

Vantagens: não exige modificação na topologia da rede.

Desvantagens: não é qualquer roteador que possui suporte para o netflow. É necessário uma máquina para fazer a coleta e análise das mensagens. O netflow não analisa o conteúdo dos pacotes/fluxos.

Veremos mais detalhes sobre o Netflow mais adiante.

Port Accounting com iptables

O iptables é o comando utilizado para manipular as regras de um firewall Linux - o netfilter, presente no kernel > 2.4. O netfilter possui uma série de contadores internos que contabilizam quantos pacotes/bytes passaram por cada chain/regra do firewall. Podemos utilizar então esses contadores para contabilizar e caracterizar o tráfego de rede.

Como normalmente criamos regras baseadas em origem/destino e porta de origem/destino chamamos essa seção de "Port Accounting", mas o termo mais correto seria "Rule Accounting", pois iremos contabilizar pacotes baseados em regras.

Vamos partir do princípio que o leitor tenha alguma familiaridade com o netfilter/iptables. Caso não tenha consulte o documento <http://lyra.soueu.com.br/tiki-index.php?page=LinuxRouter> para uma introdução ao assunto.

Utilização básica

Para obtermos informações sobre as regras e quantidade de bytes e pacotes que passaram por cada uma delas, basta utilizar o comando:

```
# iptables -L -v -x
```

Esse comando terá como saída as chains/regras da tabela filter, como no exemplo abaixo:

```
Chain INPUT (policy ACCEPT 314671 packets, 24706300 bytes)
  pkts      bytes target     prot opt in     out     source
on

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts      bytes target     prot opt in     out     source
on

Chain OUTPUT (policy ACCEPT 313030 packets, 108905020 bytes)
  pkts      bytes target     prot opt in     out     source
on
```

No exemplo podemos verificar a quantidade de pacotes e bytes que entraram e saíram da máquina (chains INPUT e OUTPUT). Veremos a seguir como criar regras e detalhar mais esse tráfego.

Criando Regras

A criação de regras para a contabilização não é muito diferente da criação de uma regra para um firewall qualquer. A diferença fica por conta do fato que nesse caso estamos interessados em regras que permitam que o tráfego passe, mas antes seja "classificado". Veremos alguns exemplos a seguir.

Contabilizando o tráfego SMTP para a máquina local:

```
# iptables -A INPUT -p tcp --dport 25 -j ACCEPT
# iptables -A OUTPUT -p tcp --dport 25 -j ACCEPT
```

Contabilizando o tráfego www para a rede 192.168.0.0/24:

```
# iptables -A FORWARD -p tcp -d 192.168.0.0/24 --sport 80 -j ACCEPT
```

Contabilizando o tráfego

Já sabemos como separar e contar o tráfego utilizando regras, e como mostrar os valores dos contadores do iptables. Vamos ver agora algumas idéias de como facilitar o tratamento e contabilização da saída do comando "iptables -L -v -x". Vamos trabalhar com a seguinte situação: desejamos saber quanto "download" é feito de nosso servidor www. Para isso poderíamos criar no servidor a seguinte regra:

```
# iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT
```

A saída do comando iptables -L -v -x seria algo como:

```
Chain INPUT (policy ACCEPT 295416 packets, 309520165 bytes)
  pkts      bytes target     prot opt in     out     source            destinati
on
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts      bytes target     prot opt in     out     source            destinati
on
Chain OUTPUT (policy ACCEPT 193921 packets, 112854141 bytes)
  pkts      bytes target     prot opt in     out     source            destinati
on
  17179 16049305 ACCEPT     tcp  --  any    any    anywhere          anywhere
      tcp spt:www
```

Poderíamos criar agora um script que leia essa saída e nos mostre o número de bytes da regra que nos interessa, como por exemplo:

```
#!/bin/sh

saida=$(iptables -L -v -x | awk '/spt:www/ {print $2}')
echo $saida
```

Poderíamos incluir esse script na configuração do daemon snmp e assim colhermos essas estatísticas remotamente, utilizando por exemplo o Cacti, que foi visto no capítulo anterior. Veja mais detalhes sobre o daemon snmp no documento <http://lyra.soueu.com.br/tiki-index.php?page=LinuxRouter>.

Sniffers

Iremos tratar aqui de softwares que capturam e analisam tráfego de uma rede. Antes de falarmos dos principais sniffers, iremos verificar uma funcionalidade que o kernel dos principais sistemas operacionais disponibilizam: Berkeley Packet Filter - BPF.

BPF - BSD Packet Filter

O BPF é uma facilidade para capturar pacotes em user-level, permitindo o uso de estações de trabalho para monitoramento da rede de forma eficiente. Como monitores de rede rodam no espaço de processos user-level, os pacotes devem ser copiados para um limite permitido para que possam ser manipulados pelos monitores. Esta cópia é feita por um agente do kernel chamado packet filter. O BPF permite que essa cópia seja feita de forma muito eficiente já que não funciona sobre stack-based como os antigos packets filters, mas funciona baseado num novo registro e com um bom sistema de bufferização que permite que seja muitas vezes mais rápido.

BPF no GNU/Linux

No Linux temos uma derivação do BPF, chamada Linux Socket Filtering - LSF. O LSF tem exatamente a mesma estrutura do BPF, mas o LSF é muito mais simples.

Agora que já sabemos a parte do kernel para o packet filter, iremos precisar de uma biblioteca que faça a interface user-level, a libpcap.

libpcap

A libpcap é uma interface independente de sistema para capturar pacotes em user-level. A libpcap suporta mecanismos de filtragem baseados em BPF. Iremos tratar adiante dos principais sniffers que suportam a libpcap, como o tcpdump, ethereal, ngrep, etc.

tcpdump

O tcpdump é uma poderosa ferramenta de monitoramento e sniffer de rede que suporta muitos protocolos como ICMPv4, IPv6, ICMPv6, UDP, TCP, SNMP, AFS, BGP, RIP, PIM, DVMRP, IGMP, SMB, OSPF, NFS e muitos outros tipos. O tcpdump imprime o cabeçalho de pacotes que casam com uma expressão booleana.

Abaixo temos alguns exemplos de utilização do tcpdump.

Para imprimir todos os pacotes da interface eth1 sem resolução de nomes de hosts e portas:

```
# tcpdump -n -i eth1
```

Para imprimir tráfego do host diablo:

```
# tcpdump host diablo
```

Para imprimir todos os pacotes IP's entre a maverick e qualquer host exceto a galaxie:

```
# tcpdump ip host maverick and not galaxie
```

Para imprimir todo tráfego entre a rede local e a rede 192.168.0.0/24:

```
# tcpdump dst net 192.168.0.0/24
```

Para imprimir todos os pacotes TCP SYN e FIN que não envolvam hosts locais:

```
# tcpdump 'tcp[tcpflags] & (tcp-syn|tcp-fin) != 0 and not src and dst net localnet'
```

Para imprimir todos os pacotes ICMP que não são echo requests/replies:

```
# tcpdump 'icmp[icmptype] != icmp-echo and icmp[icmptype] != icmp-echoreply'
```

Como podemos ver o tcpdump permite muitas expressões, que podem ser melhor esclarecidas no manual.

ethereal

O Ethereal é um analisador de tráfego de rede (sniffer). Ethereal possui uma interface GUI e captura pacotes no formato libpcap.

O ethereal mostra 3 janelas de visualização de um pacote:

- Uma linha sumarizada

- Uma árvore do protocolo
- Hex Dump mostrando exatamente o conteúdo do pacote

ngrep

O ngrep permite que seja especificado uma expressão regular para casar com dados de pacotes. Atualmente o ngrep reconhece: TCP, UDP e ICMP sobre Ethernet, PPP, SLIP e null interfaces. O ngrep também entende os filtros BPF.

Exemplo:

irá imprimir na saída padrão, o conteúdo dos pacotes que casarem com a string kaza (ignore case):

```
# ngrep -l -q -t -d eth0 -i 'kaza'
```

iptraf

O iptraf é um monitor de rede baseado em ncurses que gera várias estatísticas de rede incluindo informações TCP, UDP, ICMP, OSPF, ethernet load, etc.

Ntop

O Ntop - Network Traffic Probe - é uma aplicativo para Linux capaz de mostrar a utilização da rede detalhando a utilização por host, protocolo, etc. O Ntop possui uma interface web e é capaz de gerar gráficos o que facilita a interpretação de estatística de uso. Veremos a seguir como instalar e configurar o NTop.

Pré-requisitos

Para compilar e instalar o Ntop será necessário ter instalado o seguinte:

- autoconf, automake
- openssl, openssl-dev
- gdbm, gdbm-dev
- libpcap

Obtendo e Instalando o Ntop

O site oficial do Ntop é o <http://www.ntop.org>, mas os fontes do programa devem ser obtidos a partir da página do projeto no sourceforge: <http://sourceforge.net/projects/ntop>. Quando esse documento foi escrito a versão estável era a 2.2c. Baixe o arquivo e descompacte-o:

```
# tar xvfz ntop-2.2c.tgz
```

Será criado o diretório ntop e dentro dele haverá dois diretórios, um contendo o programa propriamente dito e outro contendo bibliotecas necessárias para a compilação do programa. Vamos começar compilando as bibliotecas:

```
# cd gdchart0.94c  
# ./buildAll.sh
```

Feito isso vamos configurar e compilar o Ntop:

```
# cd ../ntop  
# configure --prefix=/usr/local/ntop
```

Caso não haja nenhum erro fatal na configuração, compile e instale o Ntop com:

```
# make  
# make install
```

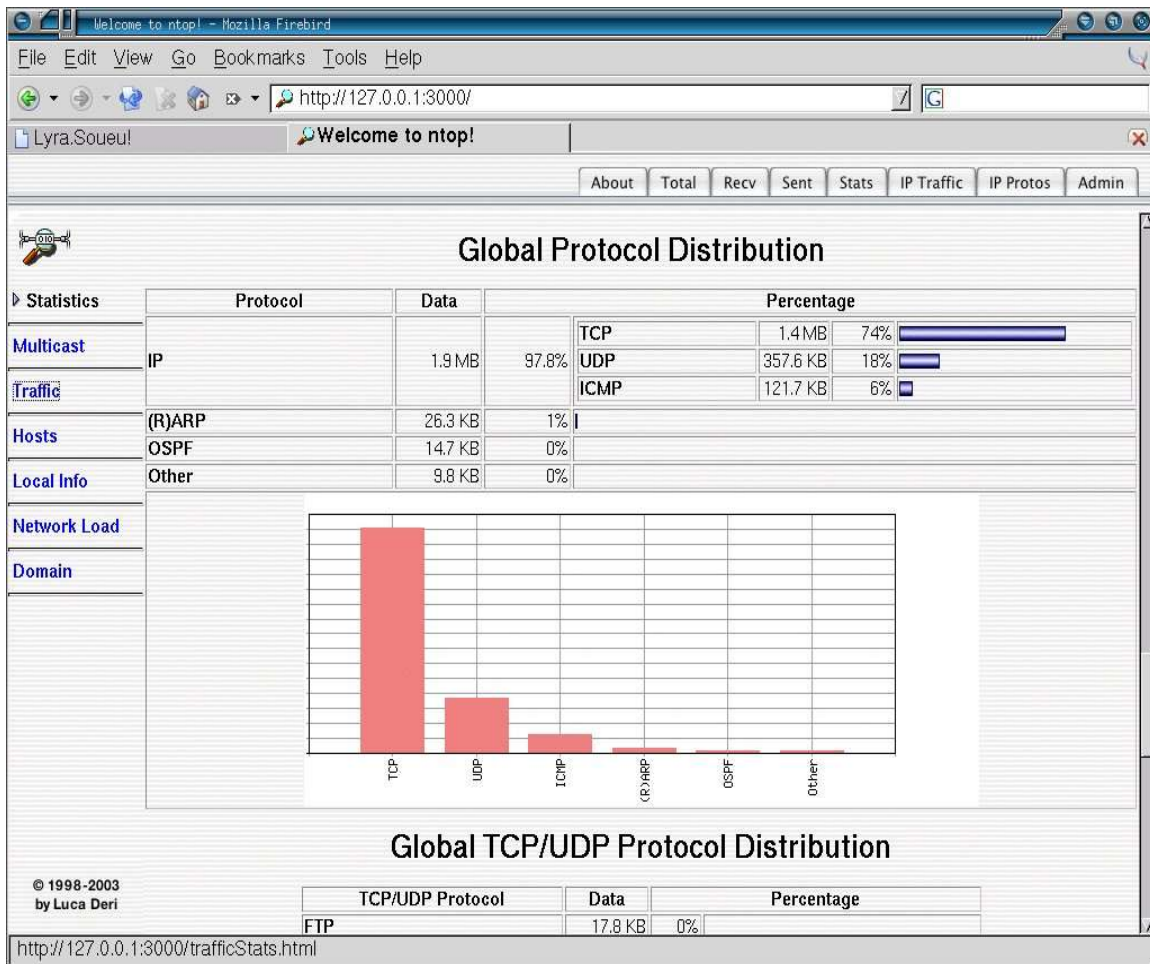
Finalmente crie dentro do diretório /usr/local/ntop, o diretório var/ntop:

```
# mkdir -p /usr/local/ntop/var/ntop
```

Utilizando o Ntop

O Ntop não possui um arquivo de configuração, logo se você deseja utilizar alguma configuração diferente dos valores padrões será necessário passar argumentos via linha de comando ou criar um pequeno script que faça isso. Rode o Ntop com a opção -h para obter uma lista de parâmetros que podem ser utilizados na linha de comando.

A utilização básica do Ntop é bastante simples, basta iniciar o Ntop e em seguida acessar o endereço <http://127.0.0.1:3000> no seu navegador. Na primeira vez que o Ntop é executado ele irá solicitar uma senha de administrador, que poderá ser utilizada depois para configurar alguns parâmetros via interface web. Veja abaixo um exemplo de estatística que o Ntop é capaz de mostrar:



Como o Ntop faz o acompanhamento de cada fluxo que chega a ele, é normal o Ntop consumir grande quantidade de memória. Quanto maior a rede, mais memória o Ntop irá consumir. Algumas coisas podem ser feitas para evitar que isso aconteça, entre elas: desabilitar o acompanhamento de sessões TCP (parâmetro -z) e diminuir o tempo que uma sessão deve permanecer inativa para ser retirada da memória. Para fazer isso é necessário alterar o arquivo `globals-define.h` e recompilar o Ntop.

Netflow

A tecnologia [NetFlow](#) desenvolvida pela Cisco, permite uma medição eficiente para um conjunto chave de aplicações incluindo:

- Contabilização de tráfego de redes.
- Tarifamento baseado em utilização de rede
- Planejamento de redes
- Detecção de DOS e DDOS
- Dentre outras possibilidades.

[NetFlow](#) exporta dados, mas estes dados estão condensados e prove uma forma de se analisar os dados através uma aplicação.

Para o SO GNU/Linux existem algumas ferramentas que geram os probes, ou seja, fazem o papel do [NetFlow](#), como se fossem um roteador Cisco com um IOS com suporte [NetFlow](#).

Algumas dessas ferramentas são:

- nProbe
- Flowprobe
- fprobe

Iremos tratar nesse documento de apenas um deles, o fprobe.

O fprobe exporta datagramas do [NetFlow](#) V5 para um coletor remoto, ele utiliza a libpcap o que o deixa bem poderoso para selecionar os fluxos.

Instalação

Para se instalar e configurar o fprobe é bem simples: basta baixá-lo de <http://psi.home.ro/flow/> e compila-lo.

Ou ainda para usuário Debian:

```
# apt-get install fprobe
```

Como usar

```
./fprobe -t IP:PORT [ -i interface ] [ -s scan ] [ expression ]
-t IP:PORT      NetFlow collector address
-i interface    interface to listen for traffic (default eth0)
-s scan         interval in seconds between two flow tables scans (Default: 10)
-c file         file with MAC definitions
-p             don't put the interface in promisc mode
```

```
-b          go in background (daemon mode)
-l file     log file name
expression  a bpf expression to filter traffic (See libpcap/tcpdump)
```

Exemplos:

```
# fprobe -t 192.168.0.2:9800 -i eth0
# fprobe -t 192.168.0.2:9800 -i eth0 host 10.10.10.1
# fprobe -t 192.168.0.2:9800 -i eth0 host 10.10.10.1 and (192.168.0.1 or 172.10.11.12)
# fprobe -t 192.168.0.2:9800 -i eth0 icmp[ ]0 !=8 and icmp0 != 0
```

Ferramentas para processar e gerenciar os datagramas netflows.

Existem várias ferramentas para coletar, processar e gerenciar os datagramas netflows, para o SO GNU/Linux temos:

- flow-tools
- cflowd
- ntop

Novamente iremos tratar aqui de apenas uma delas, o flow-tools.

O flow-tools é um conjunto de ferramentas para trabalhar com dados [NetFlow](#). O flow-tools suporta a versão 1, 5, 6, 8 e 14 do [NetFlow](#).

Instalação

Para se instalar o flow-tools, é simples, basta pegar o fonte em <http://www.splintered.net/sw/flow-tools/>, descompactar e compilá-lo.

Para usuários do Debian GNU/Linux:

```
# apt-get install flow-tools
```

Como usar

Existem várias ferramentas que compõem o flow-tools, veremos uma breve descrição de cada uma:

- flow-capture: Coleta, comprime, armazena e gerencia espaço em disco.
- flow-cat: Concatena arquivos de fluxos. Os arquivos de fluxos geralmente

são divididos por tempo (5 ou 15 minutos), então usamos o flow-cat para concatenar os arquivos.

- flow-fanout: Replica datagramas netflow para destinos unicast ou multicast
- flow-report: Gera relatórios para conjunto de dados Netflow
- flow-tag: Marca fluxo baseado em IP ou AS#
- flow-filter: Filtra fluxos baseado em qualquer campo exportado.
- flow-import: Importa dados de formato ASCII ou cflowd
- flow-export: Exporta dados para o formato ASCII ou cflowd
- flow-send: Envia dados sobre a rede usando o protocolo Netflow
- flow-receive: Recebe exports usando o protocolo [NetFlow](#) sem armazenar no disco.
- flow-gen: Gera dados de teste
- flow-dscan: Simples ferramenta para detectar alguns tipos de rede fazendo scanning ou DOS.
- flow-merge: Une arquivos de fluxos em ordem cronologica
- flow-xlate: Faz a tradução de alguns campos do fluxo.
- flow-expire: Expira fluxos usando a mesma política do flow-capture
- flow-header: Mostra meta informações em um arquivo de fluxo
- flow-split: Divide arquivos de fluxos para pequenos arquivos baseado em tamanho, tempo ou tags.

Abaixo veremos alguns exemplos:

Armazenar em /home/netflow/bb3 e expirar arquivos mais antigos quando atingir 3G de dados exportados por 192.168.0.1 para a porta 9800:

```
# flow-capture -w /home/netflow/bb3/ -E3G 0/192.168.0.1/9800
```

Imprimir todos os fluxos do dia 23/10/2003 das 07h00 às 07h30 da manhã:

```
# flow-cat ft-v05.2003-10-23.070000-0200 ft-v05.2003-10-23.071500-0200 | flow-print
```

Imprimir todos os fluxos do dia 23/10/2003 das 07h00 às 07h30 da manhã que entraram pela interface 1 (ver o descritor da interface via snmp) e tiveram origem na porta 80.

```
# flow-cat ft-v05.2003-10-23.070000-0200 ft-v05.2003-10-23.071500-0200 | flow-filter -i 1 -p 80 | flow-print
```

Imprimir todo os fluxos do dia 23/10/2003 das 07h00 às 07h30 da manhã que casam com a access-list flow-acl:

arquivo flow-acl:

```
ip access-list standard destino permit 200.201.0.0 0.0.255.255
```

```
# flow-cat ft-v05.2003-10-23.070000-0200 ft-v05.2003-10-23.071500-0200 | flow-  
filter -f flow-acl -D destino | flow-print
```

Imprimir todo os fluxos do dia 23/10/2003 das 07h00 às 07h30 da manhã que casam com a access-list flow-acl e fazer um relatório de IP's de destino ordenados pelo campo 2 (bytes).

```
# flow-cat ft-v05.2003-10-23.070000-0200 ft-v05.2003-10-23.071500-0200 | flow-  
filter -f flow-acl -D destino | flow-stat -S2 -f8
```

Mais exemplos podem ser vistos em:

<http://www.splintered.net/sw/flow-tools/docs/flow-tools-examples.html>

P2P

Peer-to-Peer (P2P) é uma classe de aplicações que se utilizam de recursos (ciclos de cpu, arquivos, etc) localizadas em outras máquinas conectadas à internet de uma maneira descentralizada. Normalmente associamos as aplicações P2P com aplicações que permitem a troca de arquivos pela internet como o napster, kazaa, gnutella, e-donkey, etc. E é seguindo esse enfoque que trataremos do tema aqui.

Nos últimos anos houve um crescimento acentuado das aplicações para troca de arquivos na internet. Tal crescimento tem um impacto profundo na utilização de banda em diversos backbones. Soma-se a isso o fato de que muito do conteúdo trocado entre essas aplicações seja conteúdo protegido por leis de Copyright, ou seja, "pirataria"! O tratamento desse problema tem sido particularmente difícil devido ao fato de que tais aplicações funcionam de forma descentralizada (não há um "servidor central"), e ao fato de que em muitas dessas aplicações os dois peers se comunicam utilizando portas de origem/destinos diversas. Esse último fato sozinho torna praticamente inviável a utilização de firewalls tradicionais para se combater o problema.

Já que não podemos mais utilizar simplesmente os campos dos cabeçalhos TCP/IP para identificar tais aplicações, faz-se necessário olhar o conteúdo de tais pacotes. Daí a utilização de termos como filtro de conteúdo ou filtros Nível-7 para tratar desses casos.

Veremos a seguir duas soluções utilizadas no Linux que procuram tratar dessa classe de aplicações.

L7-filter

O L7-filter é um projeto que visa criar um "classificador" de pacotes para o kernel do linux capaz de encontrar padrões, definidos por expressões regulares, nos níveis 5 a 7 dos pacotes TCP/IP. O projeto contém 3 partes: um patch para o kernel, patches para o pacote iproute (principalmente para o comando "tc") e um conjunto de padrões. Veremos a seguir como implementar essa solução em um roteador/gateway Linux. Consulte o documento <http://lyra.soueu.com.br/tiki-index.php?page=LinuxRouter> para mais informações sobre roteadores Linux.

O L7-filter funciona de maneira bastante similar ao NBAR da CISCO. Para melhorar a eficiência do filtro apenas os N primeiros pacotes de uma conexão são verificados, assim uma vez identificada a conexão não se faz mais necessário olhar todo o conteúdo dos pacotes. O valor N pode ser configurado, e o padrão são os 8 primeiros pacotes.

Preparando o Kernel

O primeiro passo para instalar o L7-filter é aplicar o patch e compilar um kernel com o suporte para o L7-filter. Não veremos aqui todo o procedimento para se compilar um kernel, somente os passos necessários para o L7-filter.

A partir da página do projeto em <http://l7-filter.sourceforge.net> podem ser obtidos os patch para kernels da série 2.4 e 2.6. Baixe o patch, e aplique-os em sua árvore do kernel. Exemplo:

```
# cd /usr/src/linux
# zcat /path/layer7-kernelpatch-v.0.4.1.gz | patch -p 1
```

Configure o kernel com um "make menuconfig" e habilite o suporte para o L7-filter em "Networking Support-> Networking Options-> QoS and/or Fair Queueing-> Packet Classifier API-> Layer 7 Classifier". Uma boa opção é utilizar uma configuração como a descrita no capítulo sobre kernel no documento citado acima. Compile e instale o seu kernel.

Tc patch

Para manipular regras com o L7-filter será necessário uma versão especial do comando tc. A partir da página do projeto pode ser feito o download do pacote iproute2 já com os patches necessários. Baixe o pacote e compile/instale com o tradicional "make; make install".

Padrões

Baixe o pacote contendo os padrões do site do L7-filter e os descompacte em um diretório de sua preferência. Esse pacote contém arquivos com o nome do protocolo (ftp, http, kazaa, etc) e uma expressão regular. Para que o novo kernel seja capaz de fazer o match desses protocolos, é necessário fazer um:

```
# cat *.pat > /proc/net/layer7_protocols
```

Criando regras com o L7-filter

A criação de regras para o L7-filter é bastante simples. Exemplo:

Para criar um regra que case com o protocolo kazaa:

```
# tc filter add dev eth0 protocol ip parent 1:0 prio 1 layer7 protocol kazaa classid 1:10
```

Mais uma vez, recomendamos a leitura do capítulo sobre Controle de Tráfego do documento <http://lyra.soueu.com.br/tiki-index.php?page=LinuxRouter>.

Obs: alguns padrões precisam observar os pacotes nas duas direções da conexão, logo pode ser necessário criar regras nas duas interfaces para que o L7-filter seja capaz de identificar as conexões corretamente.

Bandwidth Arbitrator

O Bandwidth Arbitrator foi a primeira solução de que tomamos conhecimento que abordava o problema das aplicações P2P. A motivação inicial do autor ao escrevê-la foi a de distribuir de forma mais equitativa a banda de um provedor wireless entre os seus clientes. Para conseguir isso o Bandwidth Arbitrator procura identificar os fluxos que estão consumindo mais banda e em seguida aplica uma "penalidade" a esse fluxo de maneira a atrasar os pacotes desse fluxo. A medida que a solução evoluiu muitos novos recursos foram adicionados, inclusive a capacidade de reconhecimento de fluxos utilizando o conteúdo dos pacotes, usando para isso uma parte de código emprestada do projeto L7-filter.

Uma abordagem mais profunda do Bandwidth Arbitrator foge do propósito desse documento. Estamos citando o Bandwidth Arbitrator aqui apenas para o conhecimento dos leitores e recomendamos aos interessados procurar mais informações no site do projeto em <http://www.bandwidtharbitrator.com>.

IDS

Um IDS - Intrusion Detection System - é um sistema utilizado para a detecção de tentativas de intrusão. De maneira mais precisa, estamos interessados em NIDS (Network Intrusion Detection System), que nada mais é do que um "sniffer" mais inteligente que além de observar o tráfego de rede, é capaz de reconhecer padrões de tráfego, baseados em regras, e reportar/alertar quando um padrão é reconhecido.

Embora um NIDS se encaixe muito mais como uma ferramenta de segurança de redes do que como uma ferramenta para caracterização de tráfego, estaremos falando deles aqui por que eles podem ser úteis na detecção de certos padrões de tráfego que ferramentas como o [NetFlow](#) não são capazes de detectar.

Dentre os NIDS disponíveis, um dos mais populares é o snort. Veremos como instalar e configurar o básico do snort. Veremos também uma ferramenta bastante útil para tratar os alertas gerados pelo snort chamada acid.

Snort

O snort é um NIDS bastante eficiente e com muitas opções de detecção e configuração. Veremos aqui apenas como instalar o snort, e como fazer uma configuração básica dele, ficando a cargo do leitor ler a documentação do snort para poder configurá-lo de maneira mais apropriada para cada caso.

Pré-requisitos

Para a instalação do snort proposta aqui será necessário:

- Mysql server
- pacote contendo as headers para as bibliotecas do mysql (libmysqlclient0-dev no Debian)

Obtendo e Instalando o Snort

Os fontes do snort podem ser obtidos em <http://www.snort.org/dl/>. A versão mais recente, na data que esse documento foi escrito, era a 2.0.2. Feito o download do pacote, descompacte-o:

```
# tar xvfz snort-2.0.2.tar.gz
```

Entre no diretório dos fontes, e rode o script de configuração:

```
# ./configure --with-mysql --prefix=/usr/local/snort
```

Compile e instale:

```
# make
# make install
```

Crie o diretório de configuração e de regras e copie os arquivos de configuração:

```
# mkdir /usr/local/snort/etc
# mkdir /usr/local/snort/etc/rules
# mkdir /var/log/snort
# cp ./etc/*conf /usr/local/snort/etc
# cp ./etc/*config /usr/local/snort/etc
# cp ./rules/*rules /usr/local/snort/etc/rules
```

Como pretendemos utilizar o snort logando no banco de dados mysql, será

necessário executar os seguintes passos:

```
# echo "create database snort" | mysql -p -u root
# mysql -p -u root < ./contrib/create_mysql
# mysql -p -u root
> GRANT INSERT, SELECT on snort.* to snort@localhost \
    IDENTIFIED BY 'snort_password';
```

Caso o banco de dados não esteja na mesma máquina onde está o snort, utilize `snort@maquina_snort` em vez de `snort@localhost`.

Para facilitar o processo de inicialização do snort utilize o script de inicialização que vem no pacote do snort:

```
# cp ./contrib/S99snort /etc/init.d/snort
# chmod /etc/init.d/snort
```

Edite o arquivo de inicialização de acordo com a sua instalação. Seguindo nosso exemplo devemos alterar as linhas:

```
SNORT_PATH=/usr/local/snort/bin
CONFIG=/usr/local/snort/etc/snort.conf
```

Configurando o snort

Edite o arquivo de configuração do snort (`/usr/local/snort/etc/snort.conf` no nosso exemplo) e altere as seguintes linhas:

```
var HOME_NET sua_rede/mascara
output database: log, mysql, user=snort password=snort_password dbname=snort host=localhost
output database: alert, mysql, user=snort password=snort_password dbname=snort host=localhost
```

Lembrando que essas são as configurações mínimas para colocar o snort para rodar!

Iniciando o snort

Para iniciar o snort basta usar o comando:

```
# /etc/init.d/snort start
```

O snort irá iniciar o seu trabalho de detecção de acordo com as regras configuradas. Tanto os alertas quanto o log serão guardados no banco de dados.

Veremos a seguir um front-end web que permitirá ver esses alertas/logs de maneira mais amigável.

ACID

O ACID - Analysis Console for Intrusion Databases - é um front-end web capaz de ler e interpretar os logs do snort armazenados em um banco de dados, facilitando assim a sua interpretação. Veremos a seguir como instalar e configurar o ACID.

Pré-requisitos

Para instalar o ACID será necessário ter:

- servidor web (aapche)
- php4
- Servidor mysql
- snort
- biblioteca adodb (pacote libphp-adodb no Debian)

Opcionalmente será necessário também suporte à biblioteca gd no php (pacote php4-gd2 no Debian) e a biblioteca JpGraph (<http://www.aditus.nu/jpgraph/>).

Obtendo, instalando e configurando o ACID

O ACID pode ser obtido no site <http://www.andrew.cmu.edu/~rdanyliw/snort/snortacid.html>. Baixe o pacote e descompacte-o no DocumentRoot do seu servidor web. Será necessário permitir que o ACID acesse o banco de dados do SNORT, para tanto o ideal é criar um novo usuário no mysql para esse fim. Podemos fazer isso desta forma:

```
# mysql -u root -p
> GRANT all ON snort.* TO acid@localhost \
    IDENTIFIED BY 'acid_password';
> flush privileges;
```

Podemos ter um conjunto de permissões mais restrito para o ACID. Consulte o arquivo README que acompanha o ACID para mais informações.

Edite o arquivo de configuração acid_conf.php e altera as linhas:

```
$DBlib_path = "/usr/lib/adodb";
$alert_dbname = "snort";
```

```

$alert_host      = "localhost";
$alert_port      = "3306";
$alert_user      = "acid";
$alert_password  = "acid_password";

$archive_dbname  = "snort";
$archive_host    = "localhost";
$archive_port    = "3306";
$archive_user    = "acid";
$archive_password = "acid_password";

```

Feito isso acesse o acid através do navegador web. No primeiro acesso o ACID o conduzirá à página de setup para que ele possa criar suas tabelas no banco de dados.

O ACID não possui nenhum mecanismo de restrição de acesso às suas páginas. Portanto será necessário utilizar outros modos para proteger o acesso às suas páginas, como por exemplo utilizando a autenticação pelo apache. Para proteger essa página pelo apache, crie no diretório do ACID o arquivo `.htaccess` com o seguinte conteúdo:

```

AuthName "ACID Access"
AuthType Basic
AuthUserFile /path_to_acid/htpasswd.users
require valid-user

```

E em seguida crie o arquivo de senhas e adicione um usuário:

```
#htpasswd -c /path_to_acid/htpasswd.users aciduser
```

Utilizando o ACID

A utilização do ACID é muito simples: acesse a página dele e observe os alertas e estatísticas 😊.

Conclusão

A caracterização de tráfego em uma rede não é uma tarefa trivial. A popularização de aplicações P2P contribui para tornar a tarefa mais difícil e mais necessária! Esperamos aqui ter apontado alguns caminhos ao apresentar ferramentas que permitam ao administrador conhecer mais sobre sua rede e sobre o que "circula" por ela. No mundo Linux as ferramentas estão em constante evolução, particularmente no Netfilter (código de firewall do Linux) tem surgido muitas novidades e patches que visam tratar especificamente de aplicações P2P.

Referências

Net-SNMP: <http://net-snmp.sourceforge.net>

RRDTOOL: <http://people.ee.ethz.ch/~oetiker/webtools/rrdtool/>

Cacti: <http://www.raxnet.net/products/cacti/>

Ntop: <http://www.ntop.org> e <http://sourceforge.net/projects/ntop>

Flow-tools: <http://www.splintered.net/sw/flow-tools/>

L7-filter: <http://l7-filter.sourceforge.net>

Linux Bandwidth Arbitrator: <http://www.bandwidtharbitrator.com>

snort: <http://www.snort.org>