Instituto Superior de Engenharia do Porto

FERRAMENTAS DE GESTÃO DE REDES E CLASSIFICAÇÃO DE TRÁFEGO

Estudo de um Caso

Pedro Rodrigues da Costa

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Arquiteturas, Sistemas e Redes

Orientador: Maria João Viamonte

Coorientador: André Moreira

Júri:

Presidente:

Doutor Luís Miguel Moreira Lino Ferreira

Vogais:

Doutora Maria João Monteiro Ferreira Viamonte

Mestre Jorge Manuel Canelhas Pinto Leite

Mestre André Santos Cruz Moreira

Porto, Outubro de 2012

À Minha Família e Amigos

Resumo

A gestão de redes informáticas converteu-se num fator vital para uma rede operar de forma eficiente, produtiva e lucrativa. A gestão envolve a monitorização e o controlo dos sistemas para que estes funcionam como o pretendido, ações de configuração, monitorização, reconfiguração dos componentes, são essenciais para o objetivo de melhorar o desempenho, diminuir o tempo de inatividade, melhor a segurança e efetuar contabilização.

Paralelamente, a classificação de tráfego é um tema de bastante relevância em várias atividades relacionadas com as redes, tais como a previsão de QoS, segurança, monitorização, contabilização, planeamento de capacidade de *backbones* e deteção de invasão. A variação de determinados tipos de tráfego pode influenciar decisões técnicas na área da gestão de redes, assim como decisões políticas e sociais.

Neste trabalho pretende-se desenvolver um estudo dos vários protocolos, ferramentas de gestão e de classificação de tráfego disponíveis para apoiar a atividade de gestão. O estudo efetuado terminou com a proposta e implementação de uma solução de gestão adequado a um cenário real bastante rico na diversidade de tecnologias e sistemas.

Palavras Chave: Gestão de Redes, Classificação de tráfego, Desempenho, Segurança

Abstract

The management of informatic networks has become a vital factor to a network operate efficiently, productively and profitably. The management involves monitoring and control actions in order to guarantee that the systems function properly, configuration, monitoring, reconfiguration of components, are essential actions in order to improve performance, reduce downtime, improve safety and make accounting.

At the same time, the classification of traffic is a very important issue in several activities related to the networks, such as the prediction of QoS, security, monitoring, accounting, planning backbones capacity, and intrusion detection. The variation of certain types of traffic can influence technical decisions in the area of network management as well as political and social decisions.

This work intends to develop a study of several protocols, management tools and traffic classification available to support management activity. The study carried out ended with the proposal and implementation of a management solution suitable for a real scenario, very rich in diversity of technologies and systems.

Keywords: Network Management, Traffic Classification, Performance, Security

Agradecimentos

A concretização deste trabalho só foi possível com colaboração de várias pessoas às quais gostaria de expressar a minha gratidão.

Começo por agradecer à minha família todo o apoio que me deu durante todo o processo de mestrado, pela compreensão das minhas ausências devido à minha situação de trabalhador estudante e dos 80Km que separaram estas minha duas atividades.

À minha orientadora Doutora Maria João Viamonte, que foi também professora da unidade curricular de Configuração e Gestão de Sistemas, onde são discutidos temas relacionados com a área da gestão de redes, tema da minha Tese. Durante o desenvolvimento deste trabalho orientou-me, sugerindo várias abordagens, esclarecendo dúvidas, ajudando na elaboração deste documento e fê-lo sempre de uma forma dedicada.

Ao meu coorientador Engenheiro André Moreira que me orientou e apoiou no desenvolvimento da solução de gestão apresentada para o Departamento de Engenharia Informática. Agradeço todas as sugestões e esclarecimentos dados.

Ao Engenheiro Nuno Silva, do DSI, que me ajudou a escolher o tema desta tese e a selecionar quais as ferramentas seriam interessantes de serem estudadas e abordadas. Pelos diálogos e trocas de *emails* que tivemos com os quais adquiri vários conhecimentos.

Aos Engenheiros Nuno Fonseca, Jaime Neto e Pedro Rocha da UAE - DEI (Unidade de Apoio ao Ensino do DEI), que me apoiaram no fornecimento e configuração das máquinas virtuais e computadores necessários.

Aos meus colegas de curso Ana Cerqueira, Bruno Tavares, Jorge Carneiro, Lionel Perez, João Rocha e Rute Pereira, com os quais realizei trabalhos durante o decorrer do mestrado e que também graças a eles cheguei a esta fase. Um agradecimento especial à Rute Pereira, com a qual fui mantendo contacto durante esta fase final, permitindo que nos apoiássemos mutuamente durante a elaboração das nossas teses.

Aos meus colegas de licenciatura Filipe Miranda, Pedro Tregosa e Duarte Silva, que trabalham no departamento de informática do instituto politécnico de Viana do Castelo, e que numa fase inicial da minha tese, mostraram-me as infraestruturas de rede do IPVC, ferramentas utilizadas e também me deram alguns conselhos.

Aos meus amigos.

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos

Lista de Acrónimos

ACK - Acknowledgment

ARP - Address Resolution Protocol

ASIC - Application Specific Integrated Circuit

ASN.1 - Abstract Syntax Notation version One

ATM - Asynchronous Transfer Mode

BER - Basic Encoding Rules

CMIP - Common Management Information Protocol

CMIS - Common Management Information Service

CPU - Central Processing Unit

CRC - Cyclic Redundancy Check

DAEMON - Disk And Execution MONitor

DDoS - Distributed DoS

DECT - Digital Enhanced Cordless Telecommunications

DHCP - Dynamic Host Configuration Protocol

DLC - Data Link Control

DNS - Domain Name Server

DoS - Denial of Service

ELS - Extended Link Service

FC - Fiber Chanel

FCAPS - Fault, Configuration, Accounting, Performance, Security

FDDI - Fiber Distributed Data Interface

FTP - File Transfer Protocol

GPL - General Public License

GSM - Global System for Mobile communications

HTTP - HyperText Transfer Protocol

HTTPS - HyperText Transfer Protocol Secure

IAB - Internet Architecture Board

ICMP - Internet Control Message Protocol

IETF - Internet Engineering Task Force

IMAP - Internet Message Access Protocol

IP - Internet Protocol

IPsec - Internet Protocol Security

IPX - Internetwork Packet Exchange

ISO - International Organization for Standardization

IOS - Internetwork Operating System

ITU-T - International Telecommunications Union - Telecommunication Standardization Sector

LAN - Local Area Network

LDAP - Lightweight Directory Access Protocol

MAC - Media Access Control

MbD - Management by Delegation

MIB - Management Information Base

MRTG - Multi Router Grapher

MO - Management Object

NAT - Network Address Translation

NDO - Nagios Data Out

NEB - Nagios Event Broker

NFS- Network File System

NIDS - Network Intrusion Detection System

NMS - Network Management System

NNTP - Network News Transfer Protocol

NNTPS - Nntp Protocol over TLS SSL

NSCA - Nagios Service Check Acceptor

NX-OS - Network Operating System designed by Cisco Systems

OID - Object Identifier

OSI - Open System Interconnection

P2P - Peer to peer

PBX - Private Branch Exchange

PDU - Protocol Data Unit

PEM - Privacy Enhanced Mail

PGP - Pretty Good Privacy

PNG - Portable Network Graphics

POP - Post Office Protocol

QoS - Quality of service

RAM - Random Access Memory

RFC - Request for Comments

RMON - Remote Network Monitoring

RRD - Round Robin Database

S/MIME - Secure/Multipurpose Internet Mail Extensions

SCSI - Small Computer System Interface

SCTP - Stream Control Transmission Protocol

SMTP - Send Mail Transfer Protocol

SNMP - Simple Network Management Protocol

SQL - Structured Query Language

SSH - Secure Shell

SSL - Secure Sockets Layer

SYN - Synchronize

TCP – Transmission Control Protocol

TLS - Transport Layer Security

TMN - Telecommunication Management Network

UDP - User Datagram Protocol

WAN - Wide Area Network

WEP - Wired Equivalent Privacy

1 Introdução

Os sistemas em rede são infraestruturas indispensáveis nos dias de hoje para que pessoas e organizações possam desenvolver as suas atividades.

Paralelamente, o aumento do grau de complexidade das redes e do seu tamanho exige o emprego de um sistema de gestão que proporcione qualidade de serviço, pro-atividade, diferenciação de tráfego e o suporte multifacetado de serviços, assim como integração com o processo de serviços e negócio. Esta realidade obriga à construção de mecanismos e normas de gestão mais ricos nas suas funcionalidades e adaptados aos novos cenários [Hegering, 1999] [Monteiro & Boavida, 2000][ComputerWorld, 2010][Case & Smith, 1995].

Nos próximos capítulos, o leitor terá a oportunidade de tomar conhecimento de que uma gestão efetiva terá de ser baseada no conhecimento profundo dos mecanismos de gestão, das tecnologias envolvidas, da configuração da infraestrutura e da orgânica da instituição. O estudo efetuado culminou com a implementação prática de um sistema de gestão adequado a um cenário real, bastante rico na diversidade de tecnologias e sistemas, concretamente o Departamento de Engenharia Informática do Instituto Superior de Engenharia do Porto.

1.1 Enquadramento Temático

Nunca ninguém podia prever, que a utilização da internet atingisse os níveis que se registam nos dias de hoje. Esta realidade implica que as organizações disponibilizem maiores larguras de banda. No entanto, existem disponíveis mecanismos que permitem garantir que a utilização abusiva por parte de determinados utilizadores não limite o tráfego da rede essencial para a instituição. Muitas vezes, considera-se como método mais simples para alcançar um melhor desempenho numa rede o aumento da largura de banda da mesma. Trata-se de uma solução simples já que estamos na era das Gigabits Ethernet e redes óticas, onde maiores capacidades estão prontamente disponíveis. Mais largura de banda não significa porém que se vá garantir sempre um determinado nível de performance. É possível que os vários protocolos que causam o congestionamento vão continuar a consumir a largura de banda adicional disponibilizada. A abordagem mais correta passa por analisar o tráfego através de um ponto, determinando a importância de cada protocolo e aplicação, com vista a encontrar uma estratégia que priorize o acesso à largura de banda. O protocolo FTP, por exemplo, tem uma larga tolerância a atrasos na rede ou a limitações de banda. Para o utilizador, o protocolo FTP simplesmente vai demorar mais tempo a descarregar um ficheiro. Ainda que o utilizador não fique satisfeito com esta realidade, esta particularidade do funcionamento do protocolo FTP não impede o funcionamento normal da aplicação. Por outro lado, novas aplicações de voz e vídeo são particularmente sensíveis aos atrasos da rede. Se pacotes de voz demorarem muito tempo a chegar ao destino, a consequência são os sons da conversação ficarem instáveis ou distorcidos. O QoS (*Quality of service*) pode ser usado

para fornecer serviços garantidos a essas aplicações, através da aplicação de mecanismos que vão influenciar os padrões de tráfego da rede. Aplicações críticas de negócio também podem fazer uso do QoS [Flannagan, 2001].

1.2 Objetivos e Principais Contribuições

Com este trabalho pretende-se efetuar um estudo de vários protocolos, ferramentas de gestão e de classificação de tráfego. A gestão envolve a monitorização e o controlo dos sistemas para que estes funcionem como o pretendido. Ações de configuração, monitorização e reconfiguração dos componentes são essenciais para melhorar o desempenho da rede, diminuir o tempo de inatividade da mesma, melhor a segurança e efetuar contabilização de tráfego. Paralelamente, a classificação de tráfego permite apoiar as atividades de gestão, tais como a previsão de QoS, segurança, monitorização, contabilização, planeamento de capacidade de *backbones* e deteção de invasão.

No entanto, todas estas atividades são realizadas por um conjunto de protocolos e ferramentas distintas, conduzindo a que a informação produzida por uma das ferramentas nem sempre seja utilizada pelas outras.

Este trabalho apresenta um estudo aprofundado sobre gestão de redes, protocolos e ferramentas de gestão. É também apresentado um sistema de gestão que disponibiliza de forma integrada todas estas atividades. O sistema proposto foi aplicado e implementado num cenário extremamente rico no que diz respeito à diversidade de tecnologias e sistemas.

Paralelamente, pretende-se mostrar que a disponibilização destas funcionalidades pode e deve ser compatível com os mecanismos de gestão já implementados e testados, permitindo no entanto, melhorar a função de monitorização, segurança e desempenho, em geral da rede.

1.3 Organização da Dissertação

Neste primeiro capítulo, Introdução, é feito um enquadramento temático sobre o tema deste trabalho e são referidos quais os seus objetivos bem como as contribuições dadas.

No segundo capítulo, Introdução à Gestão de Redes, são introduzidos alguns conceitos sobre a gestão de redes, é discutida a sua importância, assim como qual a contribuição dada para apoiar as decisões de gestão, fundamentais para o bom funcionamento de uma infraestrutura de rede. São descritas as áreas funcionais, arquiteturas e os seus modelos de informação e comunicação.

No terceiro capítulo, Ferramentas de Monitorização de Redes e Vulnerabilidades, são apresentados algumas ferramentas passiveis de ser usadas na monitorização de redes e caracterizadas algumas vulnerabilidades existentes nas redes, que podem colocar em causa a segurança da mesma.

No quarto capítulo, Caso de Estudo – Rede DEI - ISEP, é feita uma descrição da forma como o DEI faz a monitorização da sua rede e é apresentado o projeto proposto e implementado no âmbito desta tese, o qual tem o objetivo de melhorar a função de monitorização, segurança e desempenho desta rede.

No quinto capítulo são apresentadas as Conclusões, descritas as principais limitações deste trabalho e apresentadas direções para trabalho futuro.

2 Introdução à Gestão de Redes

2.1 Introdução e Definições

Com a cada vez maior dependência dos serviços de rede para aplicações de negócios críticas, pequenas mudanças no uso da rede podem afetar o desempenho e a confiabilidade da mesma. Isto tem um impacto direto sobre a capacidade de realizar funções de negócio e no custo da manutenção dos serviços de rede [sFlow.org, 2003].

Para que possa haver uma melhor compreensão de como realmente uma rede de computadores está a ser utilizada, podem ser usadas aplicações de gestão que permitam monitorizar a mesma. Com o uso destas será possível obter diversas informações sobre a infraestrutura, como por exemplo, quais os protocolos mais utlizados, que utilizadores ocupam uma maior largura de banda, que componentes da rede deixaram de responder, entre outras.

Mediante estas informações o gestor da rede pode tomar decisões de gestão que garantam o bom funcionamento da infraestrutura. As decisões podem ser várias, como por exemplo, bloquear acesso a certos *sites*, bloquear a utilização de alguns protocolos, limitar a largura de banda utilizada por cada utilizador, etc.

A análise e resolução dos problemas de gestão podem ter várias aproximações que vão desde o tratamento dos problemas de uma forma isolada, até a uma visão integrada da infraestrutura atuando sobre ela como um todo. É precisamente nesta última abordagem que as arquiteturas de gestão têm um papel primordial, uma vez que permitem o desenvolvimento de sistemas de gestão abertos, aplicáveis a ambientes heterogéneos existentes nas atuais infraestruturas informáticas.

2.2 Áreas Funcionais da Gestão de Redes

Para melhor definir o âmbito de gestão de redes, dados e telecomunicações foram criados vários modelos de gestão. Na Tabela 1 é possível visualizar um comparativo destes modelos [Wiki, 2012].

Modelo de gestão	Órgão responsável	Tipo de Gestão	Utilização
FCAPS	ISO	Falhas, configurações, desempenho, contabilização, segurança	Estrutura conceitual popular para gestão de redes.
TMN	ITU-T	Negócios, serviços, redes e elementos.	Estrutura conceitual popular para gestão de redes, direcionada para provedores de serviços de telecomunicações.
OAM&P	Provedores de serviço	Operação, manutenção, administração, provisionamento	Utilizado em redes de grandes provedores de serviços de telecomunicações
TOM	TeleManagement Forum	Redes e sistemas, desenvolvimento de serviços e operações, atendimento ao utilizador	Ainda em estágio conceitual
CMIP/CMIS	ISO	Desempenho, falhas, configurações	Desenvolvimento limitado, baseado em redes no modelo OSI
SNMP	IETF	Desempenho, falhas	Amplamente utilizado em redes de dados especialmente em redes baseadas em TCP/IP

Tabela 1 — Modelos de Gestão de Redes (baseado em [Wiki, 2012])

Destes modelos o que obtém mais destaque é o FCAPS, já que este serviu de base para os demais por definir as áreas funcionais da gestão de redes definidas pela OSI. O nome FCAPS é formado a partir das iniciais de cada uma das áreas funcionais de gestão (*Fault, Configuration, Accounting, Performance and Security*). Seguidamente será descrito o âmbito de cada uma destas áreas.

Gestão de configuração

A gestão de configuração reúne um conjunto de funções que coletam, monitorizam e alteram informação relativa à configuração dos sistemas de comunicação, para que seja possível efetuar alterações, sejam elas de *hardware* ou *software*. As informações recolhidas podem ser acerca da topologia da rede, versões de *software*, capacidades, etc. [Monteiro & Boavida, 2000].

Gestão de falhas

É uma das áreas funcionais de maior importância da gestão de redes já que uma das suas atividades fundamentais é a deteção de erros. Esta deteção pode ser feita através da monitorização de eventos ou ainda através de alarmes gerados por dispositivos, por exemplo, um determinado componente deixou de responder a pedidos ou um certo limite de tráfego foi ultrapassado. A deteção destes erros pode gerar relatórios (*errors log*), para que estes possam posteriormente ser consultados e analisados. Os alarmes devem desencadear ações

para que sejam tomadas medidas para a resolução do problema sejam estas feitas de forma automática ou não [Monteiro & Boavida, 2000].

Gestão de desempenho

A Gestão de desempenho está representada pelo conjunto de funções necessárias para monitorizar recursos e gerir eventos e falhas. Através dela pode ser feita uma prevenção de congestionamentos e de necessidades de crescimento das redes, para que isto seja possível é feita uma aquisição de dados aleatória na rede respeitando regras estatísticas. Alguns problemas comuns detetáveis são: aparentes tempos de resposta muito longos, aparecimento de sistemas com processamento excessivo, demora incomum no acesso a serviços, entre outros [Monteiro & Boavida, 2000].

Gestão de contabilização

Esta área é responsável pelo registo da utilização dos recursos/serviços da rede de forma a se proceder a uma taxação. Através desta pode-se verificar como a rede está a ser utilizada pelos utilizadores e a partir daí tomar decisões sobre políticas de imposição de quotas de utilização. Utilização da banda, espaço em disco, tempo de processamento, são alguns dos muitos recursos que podem ser geridos, o uso de uma forma indevida dos recursos de *email* ou de navegação também podem ser geridos pela gestão de contabilização [Monteiro & Boavida, 2000].

Gestão de segurança

Esta gestão parte por um conjunto de funções que identifiquem e protejam os dados e os equipamentos da rede de ataques e outras violações. Para que isto seja viável devem ser limitados os acessos às máquinas, bases de dados e contas de utilizador por meio de *firewall*, *proxy* e outros *softwares* de segurança. A identificação de informações e equipamentos que devem ser protegidos, como a proteção dos possíveis pontos de acesso à rede, análise diária de *logs* de servidores, efetuar *backups*, são exemplos de procedimentos que devem ser tidos em conta pela gestão. Se medidas deste tipo não forem previamente tomadas, problemas comuns poderão acontecer como é o caso de indisponibilidade de serviços, perdas de dados, ataques e vírus [Monteiro & Boavida, 2000].

2.3 Modelo Arquitetural da Gestão de Redes

Quando falamos de gestão de redes existem muitos recursos associados a este tema, mas os mais importantes são: o gestor e o objeto gerido, este último também denominado de agente (ver Figura 1) [Nogueira, 2004].

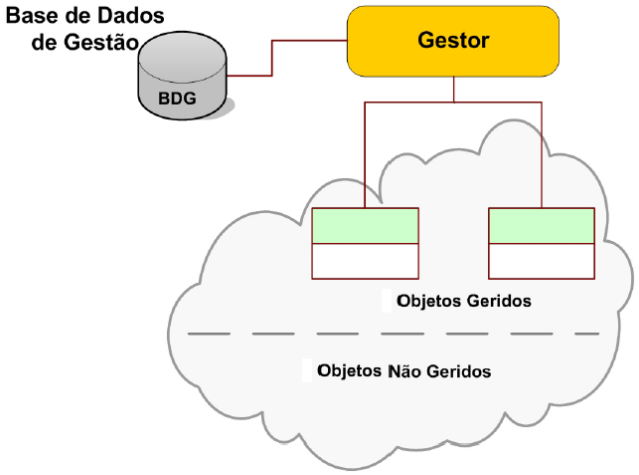


Figura 1 – Estrutura básica de gestão (baseado em [Nogueira, 2004])

O sistema gestor executa aplicações de gestão para monitorizar os objetos geridos. O objeto gerido é normalmente representado através de uma MIB (Base de informação de gestão) a qual o gestor pode consultar ou atualizar. Para que o sistema gestor e os objetos consigam comunicar e interagirem entre eles é necessário a utilização de um protocolo de gestão, este define as ações de comunicação, os seus significados e as informações transacionadas [Nogueira, 2004].

O modelo arquitetural de gestão deve traduzir a interligação entre os gestores e os objetos geridos e a forma como estes comunicam e interagem. Existem vários fatores e aspetos que realmente deverão estão envolvidos no desenho dos modelos práticos para que estes reflitam a realidade. Assim sendo para haver gestão de redes existe a necessidade de um sistema de gestão de rede (NMS) que será responsável pela gestão de um conjunto de dispositivos ou outros recursos na rede sendo eles serviços, canais, etc. Se existir a possibilidade destes dispositivos ou recursos serem representados através de uma MIB e serem acessíveis através de um protocolo de gestão, então também terão a possibilidade de serem geridos através da adição de um agente e uma MIB no dispositivo (ver Figura 2) [Nogueira, 2004].

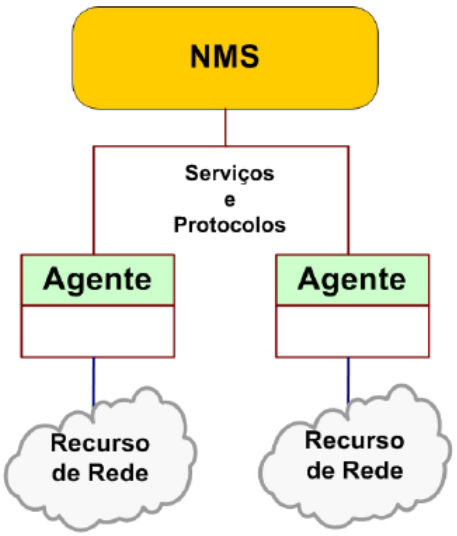


Figura 2 – Arquitetura básica de gestão de redes (baseado em [Nogueira, 2004])

Com a implementação de um protocolo de gestão e pela definição dos serviços a fornecer garante-se a comunicação entre o agente e o NMS. A interligação, interação e comunicação entre sistemas de gestão de redes (NMS) tem tido muito investimento nos últimos tempos por ser um aspeto de alta importância (ver Figura 3) [Nogueira, 2004].

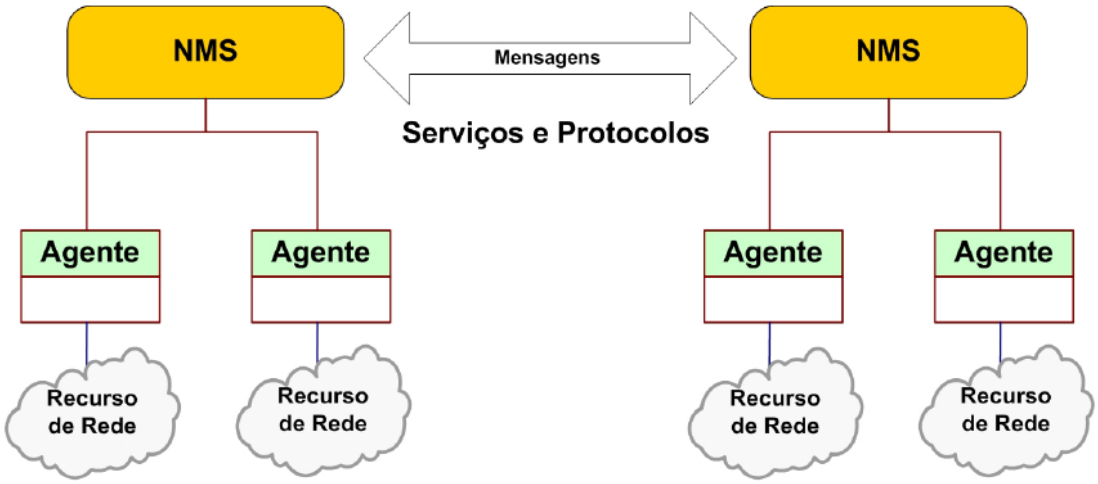


Figura 3 – Comunicação entre NMSs (baseado em [Nogueira, 2004])

2.3.1 Arquiteturas de Gestão Centralizada

As arquiteturas de gestão centralizada utilizam a topologia funcional do tipo gestor-agente utilizada pelos modelos de gestão OSI e TMN. Numa arquitetura deste tipo existe apenas uma estação gestora responsável por toda a gestão e vários objetos geridos. A interação entre estes objetos e a estação gestora é efetuada por meio de um agente sendo a

representação do objeto feita pela sua própria MIB. Os equipamentos com estas capacidades de gestão possuem estes agentes associados a cada um dos objetos geridos, a MIB permite que haja uma abstração destes. Apenas a estação gestora engloba toda a capacidade de processamento da informação de gestão, sendo que os agentes não possuem qualquer tipo de 'inteligência', estes apenas permitem a recolha remota dos dados e a execução das ações propostas pela estação de gestão. A simplicidade de desenho e implementação são duas das vantagens deste tipo de arquitetura, as possibilidades limitadas de reconfiguração, reduzida flexibilidade, não escalabilidade e quantidade de tráfego gerado são as desvantagens (ver Figura 4) [Nogueira, 2004].

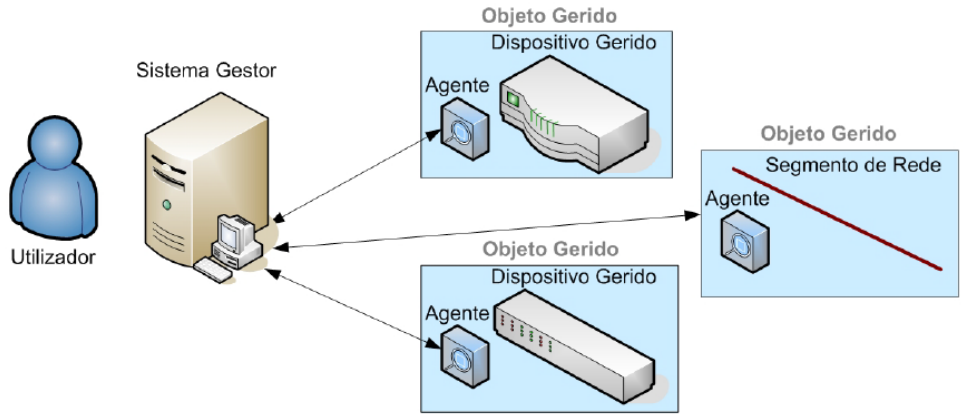


Figura 4 – Arquitetura de Gestão Centralizada (baseado em [Nogueira, 2004])

2.3.2 Arquitetura de Gestão Distribuída

No caso de redes de tamanho considerável, em que algumas podem conter milhares de dispositivos, a gestão e processamento centralizado não é uma opção muito atrativa, para estes casos, devido à necessidade de uma maior escalabilidade, a gestão distribuída é a melhor opção. Nesta arquitetura existe uma distribuição de partes das tarefas de gestão pelos diversos agentes, tendo assim cada um destes, um certo grau de 'inteligência'. Dependendo das funcionalidades pretendidas no sistema é definido o grau e o tipo de descentralização. Dado a existência de uma maior capacidade de processamento dos dispositivos de rede, a arquitetura descentralizada tem vindo a ser cada vez mais implementada. Esta arquitetura traz algumas vantagens em sistemas que precisam de manipular grandes quantidades de informação e tem como resultado um conjunto mais reduzido de dados. Melhor escalabilidade, tolerância a falhas e processamento local são ainda mais valias desta arquitetura. A gestão remota aos dispositivos também é possível, mesmo com ligação aos agentes de baixo débito. As sondas RMON (*Remote Network Monitoring*) são a forma mais simples de monitorização remota descentralizada, mas existem outras [Waldbusser, 1991] [Nogueira, 2004].

Na gestão distribuída, existe a delegação e distribuição de tarefas a outras entidades. Na gestão por delegação MbD (*Management by Delegation*) que é um exemplo de gestão

distribuída, são enviados programas de gestão para os agentes e estes executa-os. Outro exemplo de gestão por delegação é a *script* MIB. A gestão distribuída pode ainda ser dividida em arquiteturas hierárquicas, cooperantes, fracamente e fortemente distribuídas. Nestas últimas, o processamento é repartido por cada um dos agentes, são exemplos os agentes móveis e a aplicação do conceito de agentes móveis. São exemplo de arquitetura fracamente distribuída a gestão com recurso a sondas RMON. Nas arquiteturas fracamente distribuídas o processamento está concentrado num conjunto restrito de estações de gestão [Nogueira, 2004].

2.3.3 Arquitetura de Gestão Hierárquica

Este tipo de arquitetura é caracterizado por possuir uma estrutura de gestores em árvore onde a aplicação de gestão está no topo e os sistemas geridos em níveis abaixo. Para além da aplicação de gestão de topo existem aplicações de gestão nos níveis mais abaixo, as quais executam tarefas delegadas pelos gestores superiores. Esta arquitetura pode ser utilizada em redes que se encontrem dispersas geograficamente através de várias instalações, onde em cada uma existe um sistema gestor que vai recolher as informações de gestão dos objetos geridos que se encontram no mesmo local e vai armazená-las até que o sistema gestor de topo as solicite. Este funcionamento faz que haja menos tráfego na rede mas tem como desvantagem a complexidade de configuração (ver Figura 5)[Oliveira, 2005] [Nogueira, 2004].

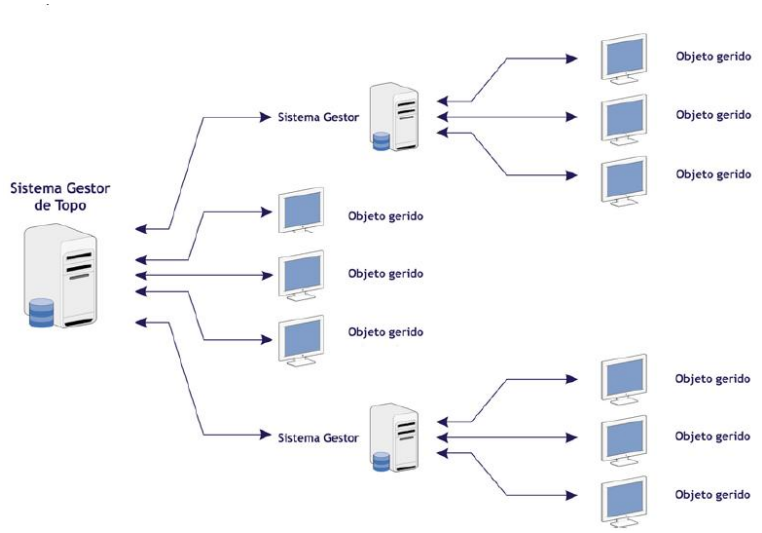


Figura 5 – Arquitetura de Gestão Hierárquica (baseado em [Oliveira, 2005])

2.3.4 Arquitetura de Gestão Cooperativa

De forma a atingir os objetivos de gestão definidos, os agentes cooperam uns com os outros usando a sua autonomia. Nesta arquitetura a estação gestora 'injeta' um programa na rede que percorre os vários ramos e assim recolhe e processa a informação. O programa irá voltar para a estação gestora contendo os resultados pretendidos (ver Figura 6) [Nogueira, 2004].

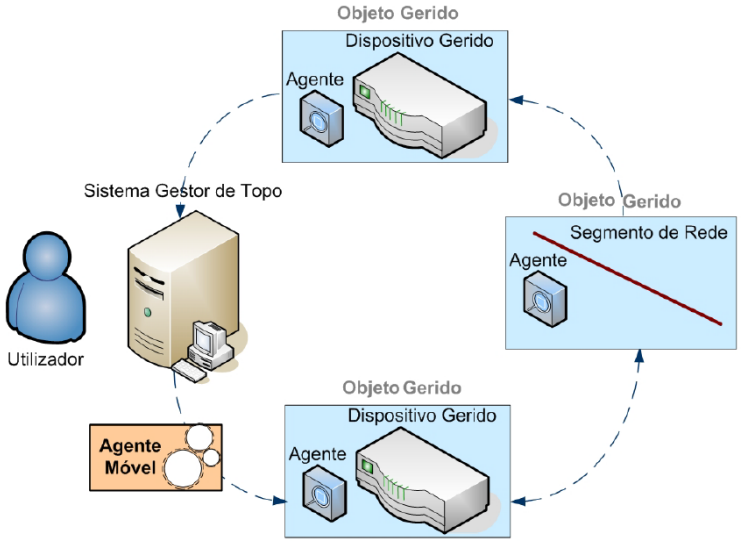


Figura 6 – Arquitetura de Gestão baseada em agentes móveis (baseado em [Nogueira, 2004])

O programa 'injetado' tem o nome de agente móvel, não tendo neste caso o termo 'agente' nada a ver com o significado habitual da palavra na gestão de redes, é um termo retirado do domínio da inteligência artificial. A cooperação pode ser limitada apenas a um itinerário de uma entidade ativa sendo possível incorporar e combinar o processamento local nos vários nós com comunicação entre eles. Esta arquitetura é a mais flexível, mas também a mais complexa. Para que os dispositivos tenham esta funcionalidade é necessário que estes possuam alguma capacidade de processamento que por vezes é bastante exigente em termos de recursos [Nogueira, 2004].

2.3.5 Arquitetura Altamente Distribuída

Na delegação de tarefas em termos verticais e funcionais esta arquitetura baseia-se na arquitetura hierárquica. Mas entre cada par de gestores é usada uma arquitetura cooperativa podendo assim haver partilha e cooperação ao nível da gestão, quer em termos complementares, quer de especialização (ver Figura 7) [Nogueira, 2004].

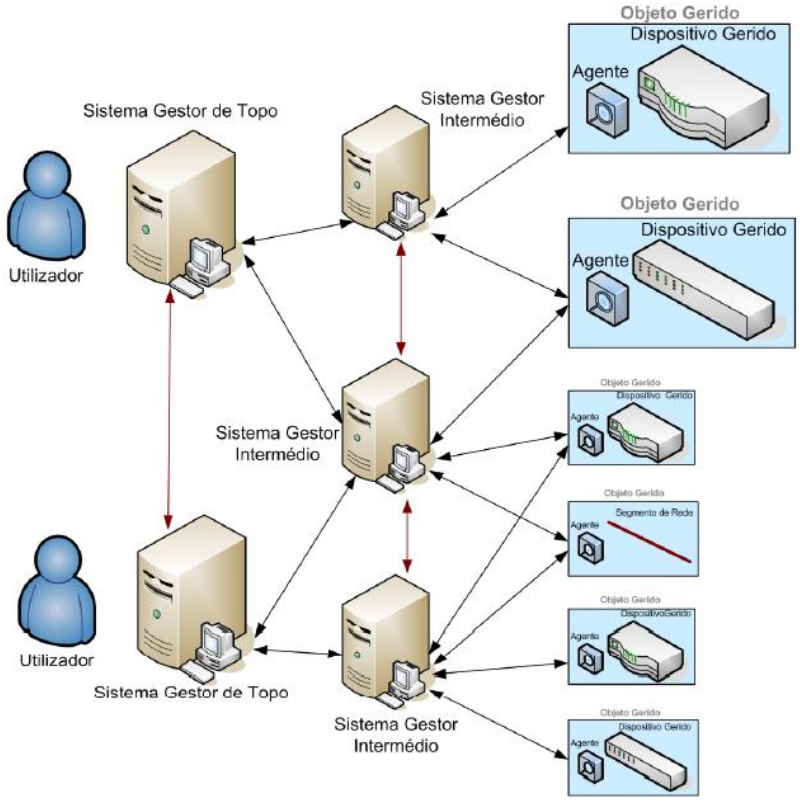


Figura 7 – Arquitetura de gestão altamente distribuída (baseado em [Nogueira, 2004])

A arquitetura de gestão altamente distribuída pode dar uma resposta com alta disponibilidade e tem uma estrutura que permite uma gestão mais escalável, permitindo uma maior desagregação das funções que por sua vez diminui a tendência de sobrecarga dos sistemas. Esta distribuição das funções pelas diversas estações pode provocar um aumento do tráfego de gestão próximo dos dispositivos geridos, já que estes podem ter que dar respostas a várias estações ao mesmo tempo. Outra desvantagem desta arquitetura está relacionada com o facto de muitas vezes os agentes dos dispositivos geridos terem de fornecer a mesma informação de gestão a várias estações gestoras. Assim haverá um incremento das tarefas a realizar, o que aumentará o tempo de processamento para a gestão e o tráfego na rede [Nogueira, 2004].

2.4 Modelo de Informação de Gestão

O modelo de informação controla os métodos usados para a modelação e descreve os objetos de gestão. Em ambientes heterogéneos, não há uma ideia comum de qual a informação que necessita ser trocada para resolver tarefas de gestão ou como os recursos devem ser caracterizados. O modelo de informação que é basicamente o coração de cada arquitetura de gestão, especifica uma descrição da *framework* dos recursos geridos. Do ponto de vista da gestão, o conhecimento de toda a estrutura de todos os processos internos dos recursos, como pontes, entidades do protocolo e sistema de ficheiros não é necessário.

Apenas os parâmetros relevantes para a gestão (como por exemplo: configuração e ajuste de parâmetros e funções, como iniciar, parar e ativar uma cópia de segurança de dados) é que devem ser modelados. Do ponto de vista da gestão pode ser interpretado como um modelo de *hardware* atual ou recursos de *software* para propósitos de gestão. Os objetos de gestão representam as características dos recursos em que a gestão atua. Consequentemente o modelo de informação deve fornecer a possibilidade de especificar como o objeto pode ser identificado, em que consiste, qual o seu comportamento, como pode ser manipulado (por exemplo, através de operações, ações ou invocações de métodos), que relação existe entre outros objetos geridos e como pode ser operado através do protocolo de gestão. A palavra gestão, do ponto de vista da arquitetura de gestão, significa acesso a um objeto gerido (MO) através de um comando de gestão, transmitido através de um protocolo de gestão para a entidade gerida (por exemplo agente) e responsável pelo objeto gerido. Monitorização constitui um acesso do tipo leitura para a representação de um MO; configuração constitui um acesso do tipo: escrita, para um MO (ver Figura 8) [Hegering, 1999].

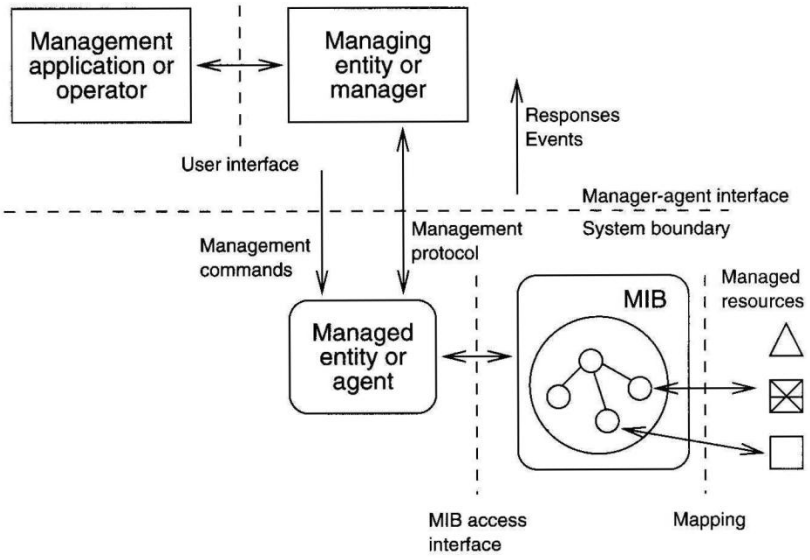


Figura 8 – Gestão através de acesso à MIB (baseado em [Hegering, 1999])

Em termos gerais, a notação *management information base* (MIB) descreve um conjunto de objetos geridos por um gestor ou um sistema gestor, embora em arquiteturas mais concretas MIB possa ter um significado ligeiramente diferente. O modelo de informação de uma arquitetura define uma aproximação de um modelo (entidade, relação, tipos de dados) e uma notação única para descrever a informação de gestão. A ISO e a IAB/IETF usam subconjuntos da linguagem de descrição ASN.1 mas definem modelos de descrição diferentes [Hegering, 1999].

2.5 Gestão Internet

Existem diversas arquiteturas onde diferentes ferramentas de gestão podem comunicar e cooperar entre si extrapolando os requisitos para uma arquitetura de gestão com os seus submodelos. Para que ferramentas de diferentes fabricantes possam integrar um sistema de gestão, existe um processo de normalização que incide sobre os seguintes submodelos: informação, organização, comunicação e funcional [Hegering, 1999].

A arquitetura de gestão que mais eficientemente incorpora estes quatro submodelos de um ponto de vista conceptual é a gestão OSI da ISO e da ITU-T, que também representa a base para a gestão de redes de telecomunicações (TMN). A OSI é a arquitetura mais amplamente utilizada no ambiente de telecomunicações no entanto, não tem um papel significativo na comunicação de dados.

A arquitetura de gestão Internet, muitas vezes também referida pelo protocolo de gestão SNMP, atualmente constitui a base para a maioria das soluções dos fornecedores de gestão na área da comunicação de dados. Uma das razões para o seu sucesso é a sua simples arquitetura. Existem ainda mais arquiteturas de gestão, como por exemplo, Corba, Web-Based, entre outras, mas por algumas vantagens já referidas e por estar mais relacionada com o objetivo deste trabalho, apenas será abordada a Gestão Internet [Hegering, 1999].

2.6 MIB - Management Information Base

A *Management Information Base* (MIB) é uma base de dados que contém informações de gestão, está organizada com um esquema de nomes único que respeita a norma ASN.1 atribuindo a cada objeto um grande prefixo. Por exemplo ***iso.org.dod.internet.mgmt.mib.ip.ipInReceives*** é um objeto que contém um número inteiro que corresponde à quantidade de datagramas IP que um determinado dispositivo recebeu. Outra forma de representar o mesmo objeto e que é usada em mensagens SNMP é feita através da atribuição de um número inteiro a cada uma das partes do nome, ficando assim representado o objeto ***ipInReceives*** por **1.3.6.1.2.1.4.3**. A MIB pode utilizar vários tipos de variáveis, inteiros, tabelas ou *arrays* e os seus valores podem ser lidos e/ou definidos por uma entidade gestora através do envio de mensagens SNMP ao agente que está num determinado dispositivo. Na Figura 9 é possível visualizar os objetos da MIB que são organizados de uma forma hierárquica, em que cada ponto do ramo da árvore está definida por um nome e um número, podendo assim as variáveis ser identificadas pela sequência de nomes e números do trajeto, desde a raiz até a um ponto específico. No topo da hierarquia está a ISO e o ITU-T, que são as duas principais entidades de padronização [Kurose, 2010].

Existe uma separação entre o protocolo de comunicação e a definição dos objetos, permitindo que sejam definidas novas variáveis MIB conforme as necessidades. Ao longo do tempo muitas variáveis têm vindo a ser criadas, algumas para protocolos, como por exemplo para o protocolo TCP, UDP, ARP e IP, outras para *hardware* de rede, como por exemplo

Ethernet, *Token Ring*, e FDDI ou ainda para alguns dispositivos, como *bridges*, *switches* e impressoras [Hegering, 1999].

A MIB-II é atualmente a especificação básica válida da informação de gestão. Nós estruturais, chamados grupos da MIB-II, foram adicionados à estrutura do nó 1.3.6.1.2.1 [Hegering, 1999].

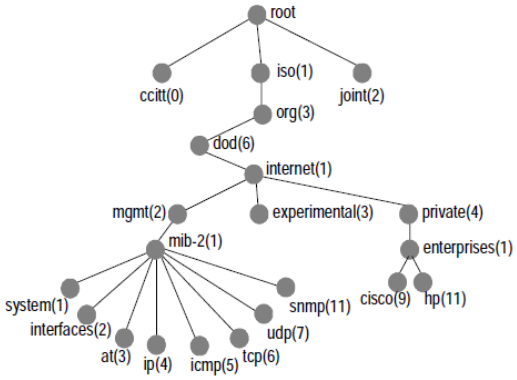


Figura 9 – Organização dos objetos da MIB (baseada em [RMON, 2003/2004])

2.7 Modelo de Comunicação Internet - SNMP

O núcleo do modelo de comunicação da internet é o *Simple Network Management Protocol* (SNMP). Até ao momento existem 3 versões a versão 1, a versão 2, com várias variantes, e a versão 3. Estas versões serão abordadas de seguida.

Na gestão Internet, os recursos também são geridos, ou seja, monitorizados e controlados através do acesso aos valores da MIB que representam os recursos. A MIB é um repositório de informação sobre os dados usados para gerir o agente. A comunicação entre o gestor e o agente é feita através do protocolo de gestão de rede SNMP. O protocolo SNMP foi desenvolvido no início dos anos 80 pelo *Internet Engineering Task Force* (IETF), sendo a sua primeira especificação o RFC1067. Este protocolo de gestão pertence à camada de aplicação do modelo OSI e com ele é possível obter informação ou alterar configurações de uma rede TCP/IP. As redes que trabalham com este protocolo têm essencialmente dois tipos de componentes, os agentes SNMP e a entidade que os gere denominada por gestor (ver Figura 10).

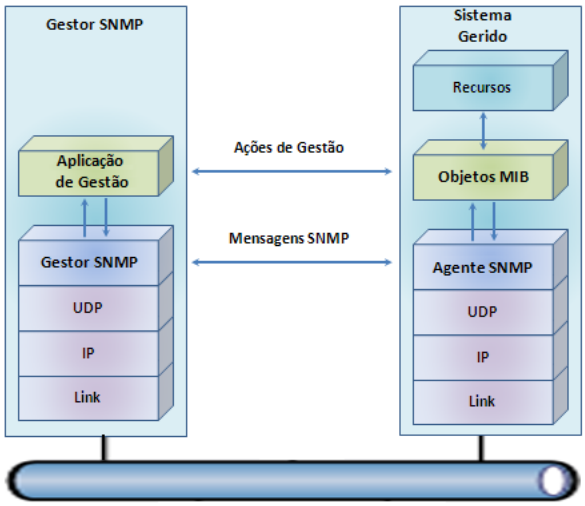


Figura 10 – Arquitetura SNMP (baseada em [André, 2011])

A gestão SNMP funciona através de uma troca de informações baseada na arquitetura cliente-servidor entre a MIB do agente e a aplicação do gestor. Utilizando o protocolo de transporte UDP (*User Datagram Protocol*) o agente realiza as ações de gestão através de pedidos aos agentes, estas ações são feitas através de quatro operações básicas, *SET*, *GET, GET-Next* e *Trap*. Um agente SNMP recebe os pedidos do gestor, realiza as ações necessárias e gera uma resposta apropriada. Durante este processo o protocolo funciona de forma assíncrona, ou seja, o gestor pode iniciar um pedido sem ter que esperar que o agente forneça uma resposta. A operação protocolar assíncrona *Trap* gerada pelo agente em situações excecionais pelo porto UDP 162, permite que o agente transmita informação ao gestor sem que tenha existido inicialmente uma solicitação, permitindo reportar eventuais acontecimentos inesperados. O SNMP fornece um total de quatro operações protocolares que podem ser visualizadas na Figura 11 [Hegering, 1999].

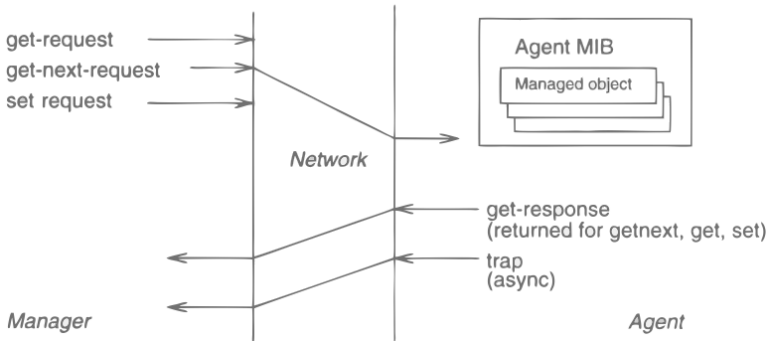


Figura 11 – Operações SNMPv1 (baseado em [Hegering,1999])

A operação *SET* conduz a uma escrita na MIB, permitindo alterar o comportamento de um recurso. A operação *GET* é utilizada para obter o valor de uma variável específica. Existe a possibilidade de se obter o valor da variável seguinte através da utilização da operação *GET-Next*, esta também pode ser usada para obter valores e nomes de variáveis de tabelas

cujo tamanho é desconhecido. O protocolo também permite que o agente informe o gestor da ocorrência de um determinado evento recorrendo (como já referido) à operação *Trap*.

O facto do SNMP usar o protocolo UDP para transporte da informação de gestão faz com que o SNMP seja pouco seguro e fiável, para além destas lacunas o facto de não ser controlado nem registado também se revela uma desvantagem. A fim de superar estes problemas o IETF desenvolveu novas versões deste protocolo sendo elas o SNMPv2 e SNMPv3 [Hegering,1999] [Pan, 1998].

Na construção, preparação, envio e receção de uma mensagem SNMP são executados os seguintes procedimentos:

1. Construir a PDU (*Protocol Data Unit*) utilizando a notação ASN.1. A PDU pode ser um '*GetRequest*', '*SetRequest*', etc;
2. Enviá-la opcionalmente para um serviço de autenticação em conjunto com os endereços de origem e destino e com o nome da comunidade;
3. Construir a mensagem SNMP, adicionando à PDU (autenticada ou não) o campo da versão e o nome da comunidade;
4. Codificar a PDU, utilizando o esquema de codificação BER (*Basic Encoding Rules*), para que seja enviada para o serviço de transporte;
5. Na receção, construir a mensagem SNMP em ASN.1 a partir de BER;
6. Validar a versão do protocolo SNMP utilizado, retirando a PDU SNMP da mensagem;
7. Enviá-la opcionalmente para um serviço de autenticação em conjunto com os endereços de origem e destino e com o nome da comunidade:
    a. Se a autenticação falhar a PDU é ignorada e é gerado um '*Trap*';
    b. Se a autenticação não falhar, enviar a PDU em ASN.1.
8. Analisar os campos da PDU, verificando se o seu preenchimento está correto;
9. Por último, executar a ação pedida, tendo em conta o perfil de comunidade identificado.

2.7.1 Polling

O gestor faz pedidos aos agentes para obter informações relativas ao comportamento do objeto gerido. Para tal, fornece o OID do objeto e aguarda uma resposta. Este processo é denominado de *polling* e através dele o gestor consegue obter os dados quando necessário. Dependendo da importância de um certo componente na rede o *polling* pode ser efetuado com maior ou menor frequência. O *polling* gera tráfego na rede e por isso o seu intervalo de tempo deve ser definido conforme a importância do objeto consultado e a carga na rede. A consulta de informação estatística pode ser efetuada apenas uma vez por dia [Muller, 1996] [Castro, 2008]. No caso de o gestor não obter uma resposta num determinado intervalo de tempo, será assumida a perda do pacote e o pedido é retransmitido um número de vezes configurável. Se todas estas tentativas falharem o gestor classifica o agente num estado de falha e despoletará os avisos configurados repetindo o processo na *polling* seguinte.

2.7.2 Interrupções

Outra forma de um gestor obter informação dos agentes é através das interrupções, os agentes estão previamente configurados para enviarem esses dados sem que o gestor os tenha solicitado. Os dados são enviados de uma forma assíncrona e assim o gestor pode ter conhecimento em tempo real de eventos pré-definidos. Comparativamente ao *pooling* este método tem algumas vantagens, já que reduz o tráfego na rede e aumenta a velocidade de resposta, ou seja o gestor vai obter a informação mais rapidamente. Desvantagens deste método estão relacionadas com facto de existir a necessidade de configurar previamente os agentes Por oposição o *polling* coloca a necessidade de configuração apenas do lado do gestor.

O gestor em função da informação de gestão obtida pode tomar decisões que conduzam à alteração da configuração dos equipamentos. Por exemplo, se for detetada a presença de um tráfego anormal num determinado interface de um *switch*, este pode ser desativado impedindo a degradação geral do funcionamento do sistema [Muller, 1996][Castro, 2008].

No protocolo SNMP a questão da segurança sempre foi apontada como um dos pontos fracos do protocolo, isto porque as primeiras versões não tiveram em consideração a questão. No entanto, as versões posteriores foram desenhadas para ultrapassar esse problema.

2.7.3 SNMPv1

Em ambientes de comunicação de dados TCP/IP o SNMPv1 é o protocolo de gestão dominante e pode ser implementado em recursos relativamente simples. Grande parte dos dispositivos suporta SNMP e existem várias aplicações para fazer a sua gestão. No entanto, existem algumas desvantagens e limitações nesta versão [Hegering, 1999]:

- O SNMPv1 é considerado um protocolo pouco seguro, é de destacar o facto de a autenticação ser feita com base numa *password* (*comunity string*), enviada em cada mensagem de uma forma imprudente (*clear text*);
- O SNMPv1 não suporta de uma forma eficiente a transmissão de grandes volumes de dados de gestão;
- O SNMPv1 apenas suporta uma configuração de *polling* que é sempre iniciada apenas pelo gestor;
- O SNMPv1 permite apenas o suporte a eventos assíncronos;
- O uso do protocolo UDP pode conduzir à perda de dados [Hegering, 1999] [Castro, 2008].

2.7.4 SNMPv2

Com vista a melhorar a primeira versão do SNMP em 1993 surgiu a versão SNMPv2, também conhecida por SNMPv2 *Classic*, ou ainda, devido ao seu modelo baseado em segurança SNMPv2p. No entanto, devido aos extensos mecanismos de segurança, as novas propostas não foram bem recebidas no mercado. De forma a tentar simplificar a proposta do grupo de trabalho da SNMPv2 foram feitos outros desenvolvimentos com as designações SNMPv2*, SNMPv2u e SNMPv2c [Muller, 1996]. Esta última versão trouxe mais funcionalidades mas continuou a usar autenticação baseada nas '*community strings*' (comparativamente à versão SNMPv1). A versão SNMPv2u é baseada num modelo de segurança orientado ao utilizador, e a versão SNMPv2* tentou apresentar uma abordagem que é um misto das versões SNMPv2p e da SNMPv2u, o que provocou confusão no mercado. Estas versões eram incompatíveis entre si fazendo com que no final a versão normalizada como sendo a SNMPv2 fosse a SNMPv2c. Trata-se de uma versão com novas funcionalidades, apesar de não ter alterações no modelo de segurança do protocolo. No entanto, o uso das restantes versões mais seguras é sempre uma opção possível [Muller, 1996].

A nova versão do SNMPv2 tem sete tipos de PDU disponíveis, os quatro existentes na primeira versão: '*GetRequest*', '*GetNextRequest*', '*GetResponse*', e '*SetRequest*' e três novos: '*GetBulkRequest*', '*InformRequest*', e SNMPv2-*Trap*. O '*GetBulk*' requer apenas uma chamada para a leitura de uma tabela (ou parte dela). O objetivo desta PDU é minimizar o número de interações do protocolo necessárias para obter uma quantidade muito grande de informação de gestão. Isto significa que necessita de apenas uma PDU em vez de uma sequência de PDUs ordenadas do tipo '*GetNextRequest*'. A PDU '*InformRequest*' é uma notificação assíncrona usada de gestor para gestor. Na versão SNMPv1 as notificações de gestor para gestor eram possíveis utilizando o *Trap*, mas como o SNMP usa UDP para o envio das mensagens a entrega destas não é garantida, isto porque os pacotes perdidos não são notificados e não há garantia da entrega. O '*InformRequest*' garante a entrega dos *Traps* enviando uma notificação do seu recebimento, através de um '*InformResponse*' que replica a informação recebida [Muller, 1996].

Nas situações em que um PDU *InformRequest* não seja entregue ao seu recetor este não enviará a resposta. Nesta situação o emissor pode retransmitir o pedido havendo assim mais probabilidade deste chegar ao destino [Cisco_IOS_11.3, 2012].

A versão final do SNMPv2 para além de gerir os recursos da rede é simples e rápida, podendo funcionar sobre TCP e outros protocolos, preferencialmente UDP, permitindo a compatibilidade com a versão SNMPv1. Embora tenha sofrido melhorias em relação à versão anterior, no que diz respeito à segurança não houve nenhuma alteração. A questão da segurança apenas sofreu uma alteração com a implementação do protocolo SNMPv3 [Muller, 1996].

2.7.5 SNMPv3

O objetivo da criação do SMNPv3 foi permitir o suporte a redes grandes e complexas variando os custos de implementação de acordo com as facilidades adicionais, aproveitando o trabalho já desenvolvido pelo SNMPv2 e sobretudo, tentar resolver o problema de segurança na operação SET que o SMNPv1 e SMNPv2 têm. Esta versão mantém o mesmo modelo de gestão de redes com os quatro componentes: O agente, o gestor, o protocolo SNMP e a MIB. Na nova versão do protocolo foram adicionadas mais duas entidades sendo o motor SNMP designado por '*snmpEngineID*' e aplicações SNMP. A função do motor SNMP é enviar, receber, autenticar, encriptar as mensagens e controlar o acesso aos objetos geridos. Dentro deste motor SNMP estão incluídos os seguintes componentes:

- *Dispatcher* - verifica a versão do protocolo SNMP de cada mensagem recebida, no caso de suportar a versão a mensagem é enviada para o modelo de processamento de mensagem;
- *Message Processing Subsystem*: responsável por preparar as mensagens que vão ser enviadas e pela extração de dados das mensagens recebidas;
- *Security Subsystem:* responsável pela autenticação e encriptação das mensagens.
- *Access Control Subsystem*: controla o objeto gerido [Muller, 1996].

A administração remota para o SNMPv3 suporta novos protocolos de segurança que são definidos como módulos separados de modo a permitir o seu crescimento. O SNMPv3 é um protocolo baseado em *standards* de interoperabilidade para gestão de redes. Tem a vantagem de providenciar acesso aos equipamentos com segurança através da combinação de autenticação e encriptação de pacotes na rede [Systems, 2012]. Os procedimentos seguros desta versão são: Autenticação e Integridade da mensagem, confidencialidade e controlo de acesso. O SNMPv3 garante a segurança que os dados não sofreram modificações, sendo que as informações mais delicadas, como os pacotes de configuração de dispositivos SNMP são encriptadas.

2.8 RMON - Remote Network Monitoring

A MIB RMON tem a capacidade de analisar o funcionamento da rede em geral e também da influência desta nos seus componentes individuais sendo eles computadores, *switchs* ou *routers*. Atualmente existem duas versões a RMONv1 e a RMONv2 inicialmente especificadas nos RFC1271 e RFC2021 respetivamente. A primeira versão disponibiliza dados estatísticos ao nível dos pacotes e a segunda ao nível de rede e da aplicação. Na Figura 12 é possível visualizar os grupos definidos na MIB RMONv1 [ehrizo,2011] [tcpipguide,2011].

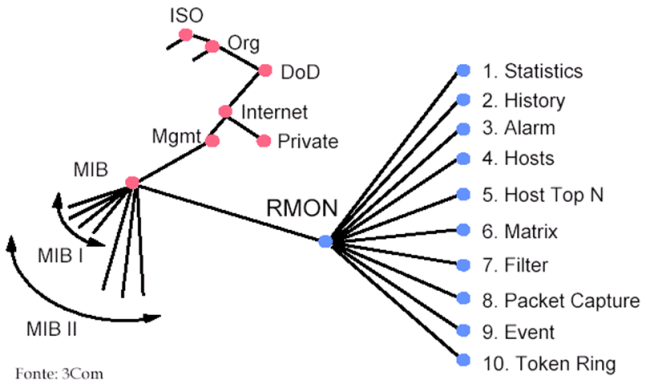


Figura 12 – Identificadores dos objetos RMON (baseada em [RMON2, 2011])

O OID do RMONv1 é 1.3.6.1.2.1.16 (iso.org.dod.internet.mgmt.mib-2.rmon). A Tabela 2 apresenta uma descrição dos grupos que a constituem.

Grupo	**Descrição**
statistics 1.3.6.1.2.1.16.1	Estatísticas em tempo real de cada interface Ethernet. Utilização, colisões, erros CRC.
history 1.3.6.1.2.1.16.2	Registo de amostras periódicas dos dados estatísticos do grupo '*statistics*'.
alarm 1.3.6.1.2.1.16.3	Definições das *Traps* SNMP RMON que serão enviadas quando uma variável monitorizada ultrapassar os limites estabelecidos, para isso, o grupo '*alarm*' realiza amostragens.
hosts 1.3.6.1.2.1.16.4	Estatísticas sobre bytes enviados/recebidos, *frames* enviadas/recebidas, MAC de origem/destino. Estes dados são recolhidos dos *hosts* que são descobertos na rede através da captura de pacotes recolhidos de uma forma promíscua.
hostTopN 1.3.6.1.2.1.16.5	Relatórios das conexões mais ativas num intervalo de tempo especificado.
matrix 1.3.6.1.2.1.16.6	Armazena tráfego enviado/recebido entre dois endereços, criando uma nova entrada na sua tabela quando deteta uma nova conversação.
filter 1.3.6.1.2.1.16.7	Define um padrão para um pacote de dados e através de um filtro pode recolher dados ou gerar um evento.
capture 1.3.6.1.2.1.16.8	Recolhe e reencaminha os pacotes que correspondem a um filtro do grupo '*filter*'.
event 1.3.6.1.2.1.16.9	Envia alertas (*Traps* SNMP) para o grupo '*alarm*'

Tabela 2 — Objetos da RMONv1 (baseada em [ehrizo,2011] [tcpipguide,2011])

A RMON1 realiza maioritariamente uma análise de protocolo nos níveis OSI 1 e 2, a RMON2 (RFC 2021) estende essa análise para incluir dos níveis 3 ao 7 (ver Figura 13). Esta extensão de análise melhora o controlo da monitorização Internet, bem como do nível de aplicação (e-mail, transferência de ficheiros, WWW) porque a informação importante para o gestor já é processada no agente RMON, em outras palavras em agentes remotos (ver Figura 14) (ver Apêndice A) [Networkers, 2005][Muller, 1996].

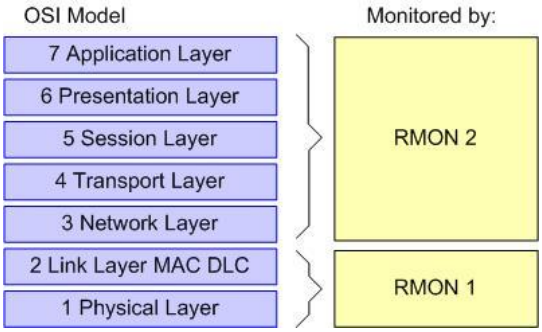


Figura 13 – Abrangência RMON1 e RMON2 (baseado em [RMON, 2011])

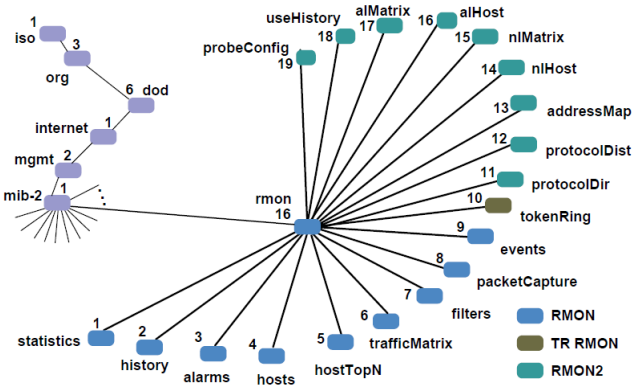


Figura 14 – Identificadores dos objetos RMON1 e RMON2 (baseado em [RMON, 2011])

A Tabela 3 apresenta uma descrição dos grupos que constituem a RMONv2.

Grupo	**Descrição**
Protocol Directory 1.3.6.1.2.1.16.11	Especifica uma lista de protocolos que o elemento RMON pode monitorizar.
Protocol Distribution 1.3.6.1.2.1.16.12	Contém estatísticas (bytes, pacotes) de tráfego relativo a cada protocolo.
Address Map 1.3.6.1.2.1.16.13	Faz o mapeamento entre endereços IP e endereços MAC.
Network-Layer Host 1.3.6.1.2.1.16.14	Contabiliza o tráfego enviado/recebido relativo ao protocolo de rede.
Network-Layer Matrix 1.3.6.1.2.1.16.15	Contabiliza o tráfego por cada par origem/destino relativo ao protocolo de aplicação.
Application-Layer Host 1.3.6.1.2.1.16.16	Contabiliza o tráfego enviado/recebido relativo ao protocolo de rede
Application-Layer Matrix: 1.3.6.1.2.1.16.17	Contabiliza o tráfego por cada par origem/destino relativo ao protocolo de aplicação.
User History 1.3.6.1.2.1.16.18	Recolhe amostras periódicas sobre variáveis específicas do utilizador.
Probe Configuration 1.3.6.1.2.1.16.19	Configuração dos parâmetros de operação da RMON.
RMON Conformance	Requisitos para a conformidade RMON2.

Tabela 3 — Grupos da RMONv2

3 Ferramentas de Monitorização de Redes e Vulnerabilidades

3.1 Introdução

Existem vários fabricantes que disponibilizam plataformas de gestão e monitorização de redes mas por vezes estas requerem muito tempo de implementação e são muito dispendiosas. Normalmente estas plataformas têm um bom nível de integração podendo ser configuradas de forma a satisfazer as características pretendidas. No entanto, grande parte delas possui mais ferramentas/funcionalidades do que as realmente necessárias. Por outro lado, os produtos individuais têm um preço mais reduzido mas podem não satisfazer todas as necessidades presentes e futuras. Estas aplicações podem ser ativas ou passivas, podendo as primeiras ser configuradas para assumirem ações automatizadas, a serem usadas em tarefas mais repetitivas, como por exemplo reiniciar máquinas ou serviços. No entanto, estas aplicações consomem mais capacidade de processamento e espaço em disco. Para a monitorização de tráfegos e tempos de resposta nos dispositivos, são normalmente mais utilizadas aplicações com tecnologia passiva, podendo estas funcionar em tempo real e se necessário gerar alertas ao administrador do sistema. Várias outras variáveis podem ser monitorizadas, tais como tráfego excessivo, acessos não autorizados, etc. No entanto, normalmente nenhuma ação automática é gerada. Também existem aplicações para guardar a informação recolhida, incluindo identificações e tendências que permitem mais tarde visualizar e analisar a informação. Os relatórios gerados pelos produtos de gestão podem ser em tempo real ou históricos, sendo que os primeiros devem ajudar a resolver problemas de desempenho, preferencialmente antes que os utilizadores se apercebam de uma degradação ou falha do(s) serviço(s). Os relatórios históricos são mais utilizados para detetar tendências de utilização e planear futuras capacidades [ComputerWorld, 2010].

É de destacar que as ferramentas de monitorização e gestão após serem instaladas não ficam automaticamente operacionais. Estas têm que ser configuradas para satisfazer as necessidades de um ambiente específico. No entanto, atualmente os tempos de instalação e configuração destas ferramentas são bastante inferiores aos tempos necessários em versões anteriores, já que vão sendo integradas novas funcionalidades que facilitam o processo, como é o caso do *AutoDiscovery* que facilita o mapeamento da rede [ComputerWorld, 2010].

As plataformas podem fazer parte de uma solução completa de *hardware* e/ou *software* podendo ser soluções baseadas em agentes, ou não, sendo que se forem estes tomam a forma de código residente nos dispositivos geridos, ou perto destes, para recolher os dados. Nestes casos existe um gestor que fornece os dados recolhidos em gráficos, tabelas ou painéis de controlo customizados. As ferramentas que não usam agentes são normalmente

mais utilizadas nas funções de monitorização de dispositivos de rede e sistemas [ComputerWorld, 2010].

As ferramentas de monitorização e gestão de rede mais focadas em análise de tráfego podem, fornecer uma visão sobre os volumes e tipos de tráfego existente na rede em qualquer altura. Estas ferramentas geram alertas quando os padrões de tráfego desviam-se de um comportamento definido, o que pode indicar um problema no desempenho ou na segurança. Na análise de tráfego também podem ser usados métodos sem agentes na recolha dos dados sobre o tráfego e verificar quais os protocolos mais usados. Também é possível detetar quando estão a ser solicitados pedidos excessivos ou quando um utilizador final estiver a usar uma rede de partilha P2P [ComputerWorld, 2010].

3.2 MRTG - Multi Router Traffic Grapher

O MRTG [Oetiker's, 2011] é uma ferramenta essencialmente destinada a efetuar monitorização de dispositivos de rede SNMP. Gera gráficos no formato PNG que são inseridos em páginas HTML e que ilustram o tráfego que passa num determinado interface. O MRTG foi criado por Tobi Oetiker utiliza as linguagens Perl e C, funciona em sistemas Unix/Linux, Windows e Netware e é um *software* livre licenciado sobre a licença Gnu GPL. O MRTG também pode ser utilizado para monitorizar outros sistemas que não tenham SNMP, como por exemplo servidores de *email*, servidores Web, condições de tempo, temperatura etc, desde que haja uma fonte de dados. Os relatórios de consumo de banda HTML gerados pelo MRTG sobre um determinado interface de rede são constituídos por quatro gráficos, sendo estes: diário, última semana, últimas cinco semanas e último ano. Normalmente, trabalha-se com quatro dados na seguinte ordem: contagem de *bytes* de entrada, contagem de *bytes* de saída, *uptime* e nome. As páginas dos relatórios podem ser visualizadas através do *browser* sendo que para isso necessita de um servidor HTTP instalado [Oetiker's, 2011] [Maxwell, 1999]. Na Figura 15 é possível visualizar o funcionamento do MRTG e na Figura 16 um exemplo dos gráficos gerados.

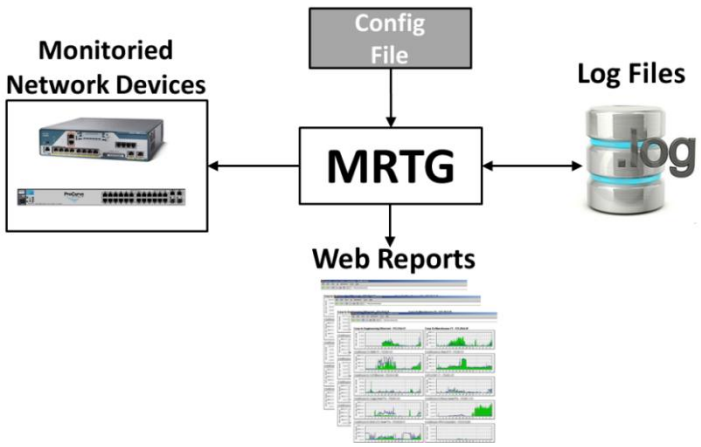


Figura 15 – Operação básica do MRTG (baseado em [Hegering, 1999])

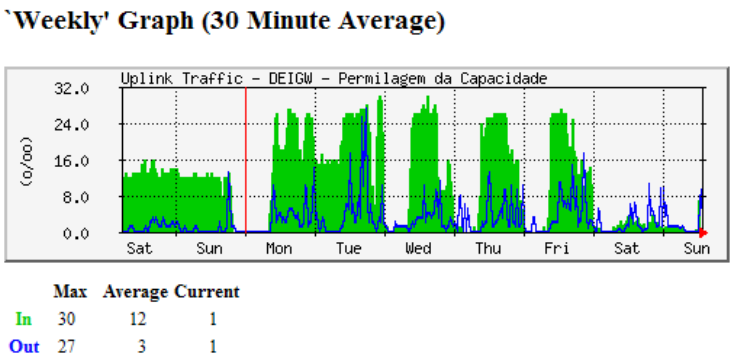


Figura 16 – Exemplo de gráfico semanal gerado pelo MRTG (baseado em [DEI, 2012])

3.3 CACTI

O Cacti é uma ferramenta para a monitorização e apresentação gráfica do tráfego de uma rede. É escrita em PHP/MySQL e usa o mecanismo *RRDtool* (Round-robin *database Tool*) [rrdtool, 2011] para guardar os dados, gerar os gráficos e coletar dados periódicos através de Net-SNMP (conjunto de aplicativos que implementam SNMP) [net-snmp, 2012].

Trata-se de uma ferramenta *Open Source* de fácil instalação e configuração, não necessita de grandes recursos, possui uma interface Web flexível e é possível partilhar *templates* com outros utilizadores. Existe a possibilidade de serem adicionados *plug-ins* para poder integrar o Cacti com outras ferramentas livres como as ferramentas Ntop ou o PHP Weathermap (ver Figura 17 e Figura 18) [Dinangkur Kundu, 2009] [Urban, 2011].

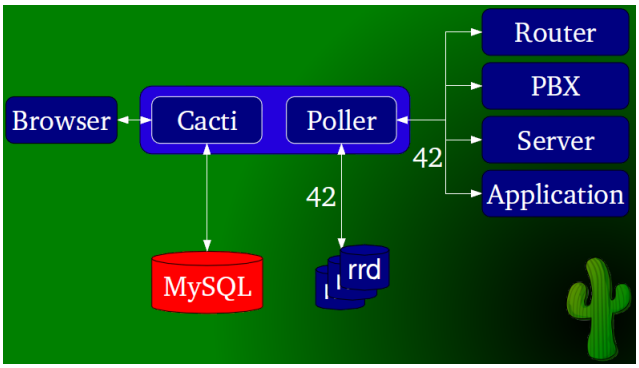


Figura 17 – Arquitetura do CACTI (baseado em [Cacti, 2009])

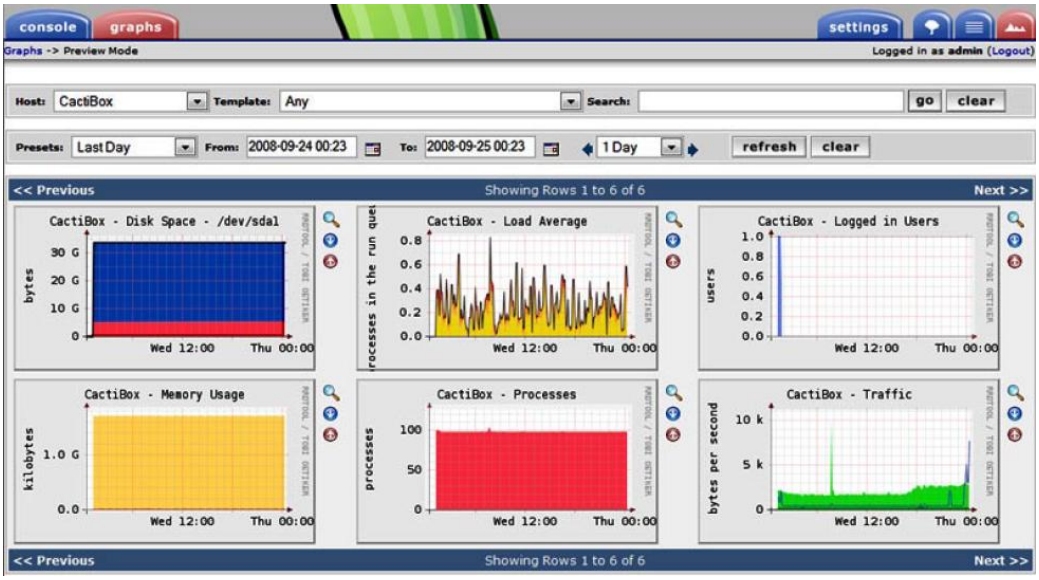


Figura 18 – Gráficos gerados pelo CACTI

3.3.1 Funcionamento do CACTI

O funcionamento do Cacti é dividido em três tarefas principais (ver Figura 19).

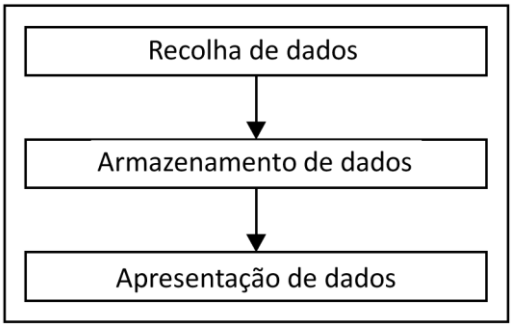


Figura 19 – Funcionamento do Cacti (baseado em [Dinangkur Kundu, 2009])

O Cacti recolhe os dados através de pedidos/respostas usando o protocolo SNMP. Existem várias formas para efetuar o armazenamento dos dados, por exemplo através de uma base de dados SQL, ou de simples ficheiros. O Cacti usa o *RRDtool* para armazenar os dados. O RRD é um sistema que armazena e mostra séries de dados temporais coletados através de diferentes dispositivos compatíveis com SNMP. Os dados históricos são consolidados e baseados em funções, tais como MÉDIA, MÁXIMO, MÍNIMO, e assim mantém o tamanho mínimo de armazenamento, este é o motivo pelo qual a tarefa de criar gráficos e relatórios a partir de ficheiros RRD se revela fácil e rápida. O Cacti usa o *RRDtool* para armazenar os dados e gerar os gráficos, mas os utilizadores não necessitam ter muito conhecimento sobre esta ferramenta, já que a criação dos gráficos pode ser feita de uma forma gráfica e sem configurações complicadas [Dinangkur Kundu, 2009].

3.3.2 RRDtool

O *RRDtool* é um sistema de registo e apresentação gráfica de alto desempenho projetado para manipular séries de dados temporais como a largura de banda de uma rede, temperatura do ambiente, carga do CPU ou para monitorizar dispositivos tais como *routers*, *ups*, etc. É uma ferramenta de base de dados *Round-robin* que é um padrão da indústria e uma solução de código aberto. O *RRDTool* é desenvolvido por Tobi Oeticker que é conhecido pela criação do MRTG, escrito na linguagem C e armazena os dados em ficheiros com extensão .rrd. O número de registos é um único ficheiro que nunca aumenta de tamanho o que significa que os registos mais antigos são frequentemente apagados. A ferramenta oferece vários comandos para aceder e manipular os ficheiros, os comandos passiveis de ser usados são: *CREATE*, *UPDATE*, *UPDATEV*, *GRAPH*, *DUMP*, *RESTORE*, *FETCH*, *TUNE*, *LAST*, *INFO*, *RRDRESIZE*, *XPORT* e *RRDCGI*. O *RRDTool* segue um processo lógico para adquirir e processar os dados adquiridos a partir das fontes de dados [Dinangkur Kundu, 2009].

3.3.3 Aquisição de dados

Quando se monitoriza um dispositivo, ou sistema, é necessário receber dados num intervalo de tempo constante. Esta tarefa não é passível de ser feita de uma forma manual e neste contexto o *RRDtool* revela-se uma ferramenta muito útil. Permitindo armazenar os dados numa base de dados do tipo *Round-robin*, dados estes que são recebidos através de *polling* num intervalo de tempo constante definido pelo administrador [Dinangkur Kundu, 2009].

3.3.4 Consolidação dos dados

O administrador do sistema pode registar os dados num intervalo de tempo de 5 minutos e estar interessado em conhecer os dados do último mês. Neste caso, o simples armazenamento dos dados num intervalo de 5 minutos durante todo o mês resolve o problema mas isso requer uma quantidade de espaço em disco rígido muito grande e também um tempo considerável para a análise dos dados. Em ambientes de redes de computadores os administradores não monitorizam apenas um dispositivo mas sim vários e o *RRDtool* ajuda nessa tarefa consolidando os dados. Criando uma base de dados Round-robin pode ser definido o intervalo de tempo ao qual a consolidação dos dados deverá ocorrer usando funções como máximo, mínimo, média entre outras [Dinangkur Kundu, 2009].

3.3.5 Ficheiros Round-robin para dados consolidados

Os valores dos dados consolidados estão armazenados em ficheiros *Round-robin*. Desta forma o *RRDTool* guarda os dados da forma mais eficiente por um período de tempo definido pelo administrador do sistema. Este processo mantem o ficheiro de base de dados com um tamanho constante para um processamento e análise mais rápidos [Dinangkur Kundu, 2009].

3.3.6 Dados não reconhecidos

O *RRDTool* armazena os dados num intervalo de tempo constante numa base de dados *Round-robin*. Por vezes esses dados não estão disponíveis para serem armazenados devido a uma falha num dispositivo ou por outro motivo. Nesse caso o *RRDTool* guarda no ficheiro RRD o dado '*UNKNOWN*' (desconhecido), este valor é suportado por todas as funções do *RRDTool* [Dinangkur Kundu, 2009].

3.3.7 Desenho de gráficos

O *RRDtool* permite que sejam gerados relatórios de uma forma gráfica ou numérica baseada nos dados armazenados nas bases de dados *Round-robin* (RRD) utilizando as suas funções de processamento gráfico. Estes gráficos podem ser personalizados em termos de cor, tamanho e conteúdo [Dinangkur Kundu, 2009].

3.4 Nagios

Existem *softwares* de monitorização de redes de computadores que têm um custo muito elevado. Considera-se um bom investimento se a ferramenta garantir o bom funcionamento da rede. No entanto, existe uma ferramenta versátil, confiável e livre que alerta o administrador sobre os eventos da rede. Essa ferramenta chama-se Nagios [Galstad, 2011], anteriormente conhecida por Netsaint, até que o seu autor Ethan Galstad, em 2002, criou as primeiras versões do Nagios 1.x. De uma forma muito célere o Nagios conquistou muitos utilizadores graças às suas funcionalidades e estabilidade [Max Schubert, 2008].

A versão 2.x incorporou novos recursos permitindo ir ao encontro das necessidades de empresas e organizações com redes maiores e mais complexas. Foi introduzido o *Nagios Event Broker* (NEB) que é uma *framework* que permite que os desenvolvedores escrevam módulos em linguagem de programação C e com estas recebam notificações de vários eventos atuando com base nestes. Ao mesmo tempo, a camada de base de dados relacional foi retirada do Nagios para que fosse feita uma distinção clara entre o núcleo do Nagios e os *add-ons/plug-ins* e para que se mantivesse o Nagios o mais flexível possível.

O NDO Utilis é um módulo baseado em NEB para o Nagios e preencheu a funcionalidade do espaço da base de dados central. Durante o ciclo de lançamento da versão 2.x o NDO Utils evoluiu e foi adotado para a visualização do NagVis [nagvis, 2011] que é um *add-on* para o Nagios muito popular [Max Schubert, 2008].

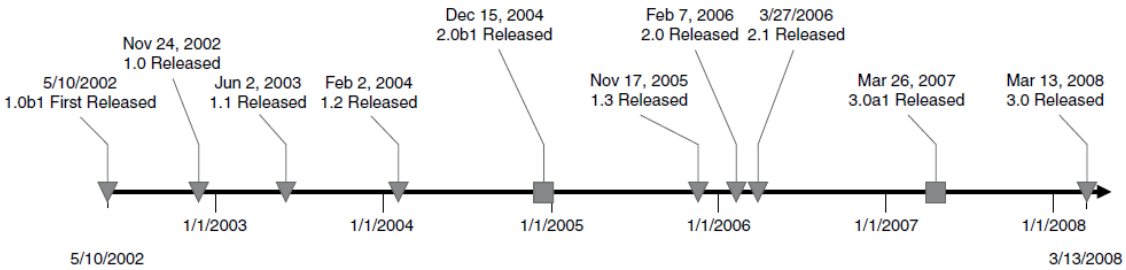


Figura 20 – Nagios *Timeline* (baseado em [Max Schubert, 2008])

Com o lançamento da versão 3.x do Nagios obteve-se o melhor das versões anteriores, manteve-se a simplicidade na configuração e simultaneamente obteve-se uma maior eficiência, facilitando o uso desta versão em ambientes mais complexos. O sistema de *templates* trouxe suporte para herança múltipla e personalizada e a disponibilidade de variáveis definidas pelo utilizador o que permitiu suporte para uma configuração mais legível. Foram adicionadas mais definições de configuração de forma que a performance do Nagios fosse mais eficiente quando este é usado com um elevado número de serviços e *hosts*. A nova interface do utilizador permite distinguir, de uma forma clara os serviços e *hosts* com problemas e os sem problemas, permitindo uma maior concentração nos serviços e *hosts* que requerem mais atenção [Max Schubert, 2008].

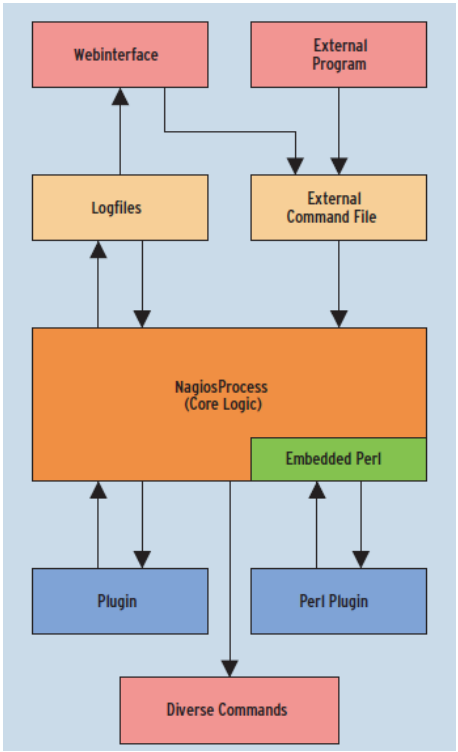


Figura 21 – Arquitetura do Nagios (baseado em [Ruzicka, 2003])

3.4.1 Arquitetura '*Plug-in*'

Os processos do Nagios invocam os *plug-ins* para coletar a informação sobre os estados dos *hosts* e serviços, estes *plug-ins* podem ser *scripts* arbitrários ou programas compilados. O estado da consulta (*Status*) está contido no código de retorno do *plug-in* (ver Tabela 4). Além disso lê a primeira linha do retorno do *plug-in* e armazena a informação recolhida em ficheiros *log*, e se necessário notifica o administrador [Ruzicka, 2003].

Código de Retorno	Estado (Status)	Significado
0	Ok	Teste completado com sucesso, o serviço está a funcionar.
1	Warning	Teste completado com sucesso, mas o resultado está fora da tolerância.
2	Critical	Teste não completado com sucesso, ou o resultado é crítico.
3	Unknown	O *plug-in* foi incapaz de realizar o teste ao serviço; o resultado é ambíguo.

Tabela 4 — Códigos de retorno dos *plug-ins* do Nagios

3.4.2 *Plug-ins* mais rápidos com o '*embedded Perl*'

O processo do Nagios é configurado com ficheiros de texto, os ficheiros centrais chamam-se: 'nagios.cfg', 'resource.cfg' e 'cgi.cfg'. O ficheiro 'nagios.cfg' aponta para outros ficheiros que contém objetos de configuração dos *hosts*, serviços e contactos. (ver Tabela 5). O administrador define objetos para representar a rede com as suas máquinas e serviços no Nagios, especificando a máquina para cada serviço (por exemplo um servidor de *email*). Cada um dos *hosts* pode ser combinado para formar um *hostgroup*, e cada *host* pode ser membro de vários grupos. Isto facilita a atribuição de um serviço para um grupo de *hosts* [Ruzicka, 2003].

Objeto	Descrição
host	Define um servidor, um computador, etc.
service	Descreve um serviço (HTTP, NFS etc.) fornecido pelo servidor.
contact	O Nagios informa esta pessoa em caso de emergência.
hostgroup	Agrupa múltiplas entradas *host* com as mesmas características num grupo. Cada *host* deve pertencer a pela menos um *hostgroup*.
contactgroup	Grupo de pessoas para permitir o envio simultâneo de mensagens.
timeperiod	Define quando um *host* ou *service* pode ser testado ou quando um *contactgroup* deve ser informado.
command	Descreve como executar os *plug-ins* e outras ferramentas externas.
host/service dependency	Define as dependências dos *hosts* e dos serviços.
host/service/ hostgroup escalation	Define o procedimento de encaminhamento de notificação.

Tabela 5 — Configuração dos objetos do Nagios (baseado em [Ruzicka, 2003])

3.4.3 Configuração estruturada

A diretiva '*parents*' (pais) permite que o administrador defina a estrutura da rede ao definir um *hos*t. Isto é muito útil porque se um *router* que interliga o servidor do Nagios a outra sub-rede for abaixo o Nagios simplesmente reporta uma falha no *router* e o sistema assinalará como inacessível qualquer *host* que tenha definido este *router* como sendo seu 'pai'.

Os administradores dos sistemas monitorizados também são agrupados como '*contactgroup*' na configuração do Nagios e um administrador pode pertencer a vários grupos. O Nagios envia mensagens de estado do sistema para os grupos e se a interface Web necessitar de autenticação, os contactos também serão utilizados como nomes de utilizador. A página Web automaticamente oculta qualquer *hosts* ao qual os contactos autenticados não devem receber mensagens. A autenticação também é necessária se houver intenção de influenciar o Nagios de algo como por exemplo impedir a verificação de um serviço, por um curto período de tempo.

O Nagios verifica os *hosts* e serviços periodicamente. O '*timeperiod*' especifica os tempos para a realização dos testes ou para o envio das mensagens para os contatos. Esta característica é extremamente útil em situações reais, um administrador pode não querer ser notificado sobre um serviço que foi abaixo fora do horário de trabalho [Ruzicka, 2003].

Para permitir que o Nagios execute comandos externos, o administrador precisa de definir objetos para esses comandos no ficheiro de configuração, onde '*command*' descreve a linha de comando incluindo todas as opções e parâmetros. Os comandos são usados essencialmente para testar os *hosts* e os serviços, mas também podem ser usados para despoletar eventos, ou mesmo enviar *emails* ou SMSs para o administrador. O Nagios possui

dois tipos de estados '*soft states*' ou '*hard states*', que podem ser definidos como '*up*', '*down*' ou '*Warning*'. Isto permite ao programa diferenciar problemas reais de avarias temporárias. Uma mudança de estado pode iniciar várias ações [Ruzicka, 2003].

3.4.4 Teste de '*hosts*' e serviços

O Nagios distingue dois tipos de testes de serviços, ativos e passivos. Os processos do Nagios executam os *plug-ins* para testes ativos em intervalos regulares como definido na variável '*normal_check_interval*'. O *plug-in* recolhe o estado do *host* ou do serviço.

Um *plug-in* em execução localmente no servidor do Nagios apenas pode verificar o comportamento externo de um *host*. Ethan Galstad programou um modo adicional chamado NRPE (*Nagios Remote Plugin Executor*), para executar *plug-ins* diretamente em *hosts* remotos. O servidor do Nagios usa o *plug-in* '*check_nrpe*' para comunicar com o *daemon* do NRPE no *host* monitorizado. Em outra alternativa o Nagios pode chamar o *plug-in* '*check_by_ssh*' para usar o protocolo SSH para executar o programa no *host*. No caso dos testes passivos, o Nagios espera um agente para informar o que descobriu, para isso usa um comando externo. Estes testes são particularmente úteis para eventos assíncronos, tais como *Traps* SNMP. Os agentes podem transferir os dados gerados por eventos deste tipo ao servidor do Nagios usando a comodidade do NSCA (*Nagios Service Check Acceptor*). O lado do cliente executa o '*send_nsca*' para enviar os resultados do *daemon* NSCA o qual está a executar o comando passivo do processo do Nagios [Ruzicka, 2003].

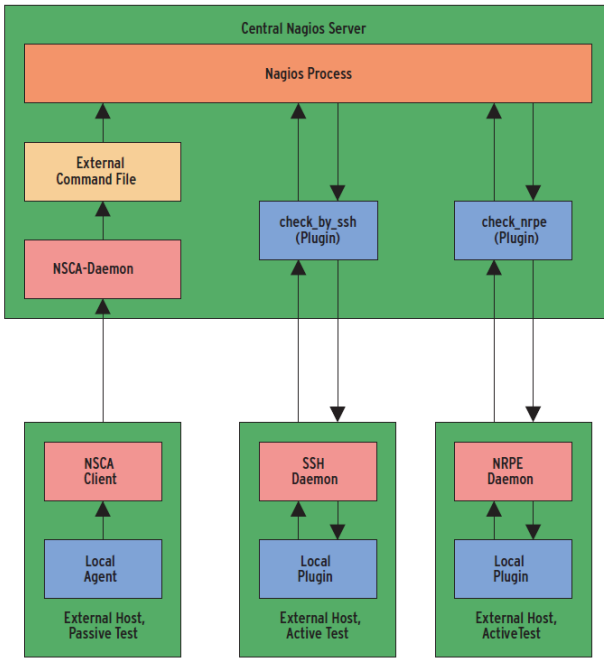


Figura 22 – Formas de efetuar testes em sistemas externos (baseado em [Ruzicka, 2003])

3.4.5 Escalonamento do tempo de inatividade

As redes de alta disponibilidade precisam de planear o tempo de inatividade, e o Nagios precisa saber quando este se verifica para evitar alertas desnecessários. Mais cedo ou mais tarde o sistema vai precisar de algum tipo de manutenção ou de uma atualização. Os sistemas que não precisam de estar disponíveis vinte e quatro horas, sete dias na semana, podem ser desligados fora do horário normal de trabalho, com sistemas de alta disponibilidade tem de se esperar por um período de inatividade. Outro novo dado relativamente ao Netsaint é o facto de que o Nagios poder escolher entre tempo de inatividade variável ou fixo. Enquanto neste último define um início e fim fixo, num tempo de inatividade variável define um período durante o qual o serviço ou *host* não estão disponíveis. Se o *host* falhar durante um período de inatividade variável, o Nagios espera por um período de tempo pré-definido antes de voltar a verificar a sua disponibilidade [Ruzicka, 2003].

3.4.6 Interface Web

O Nagios fornece uma interface abrangente oferecendo uma visão geral do atual estado da rede. Para além desta visão geral de monitorização (ver Figura 23 –), também fornece informações sobre os processos do Nagios, mostrando os testes aos *hosts* e serviços com os comentários introduzidos pelos administradores e com as horas dos próximos testes. Os relatórios completos e claros desempenham um papel importante, com eles é possível ver as mensagens transmitidas, os registos dos eventos e configuração. O resumo de estado mostra o estado do sistema completo numa tabela. O Mapa de estados (ver Figura 24) fornece vários pontos de vista sobre a estrutura da rede e mostra as dependências dos *hosts*. Esta vista apenas faz sentido para redes pequenas, porque para redes relativamente grandes rapidamente se torna confusa [Ruzicka, 2003].

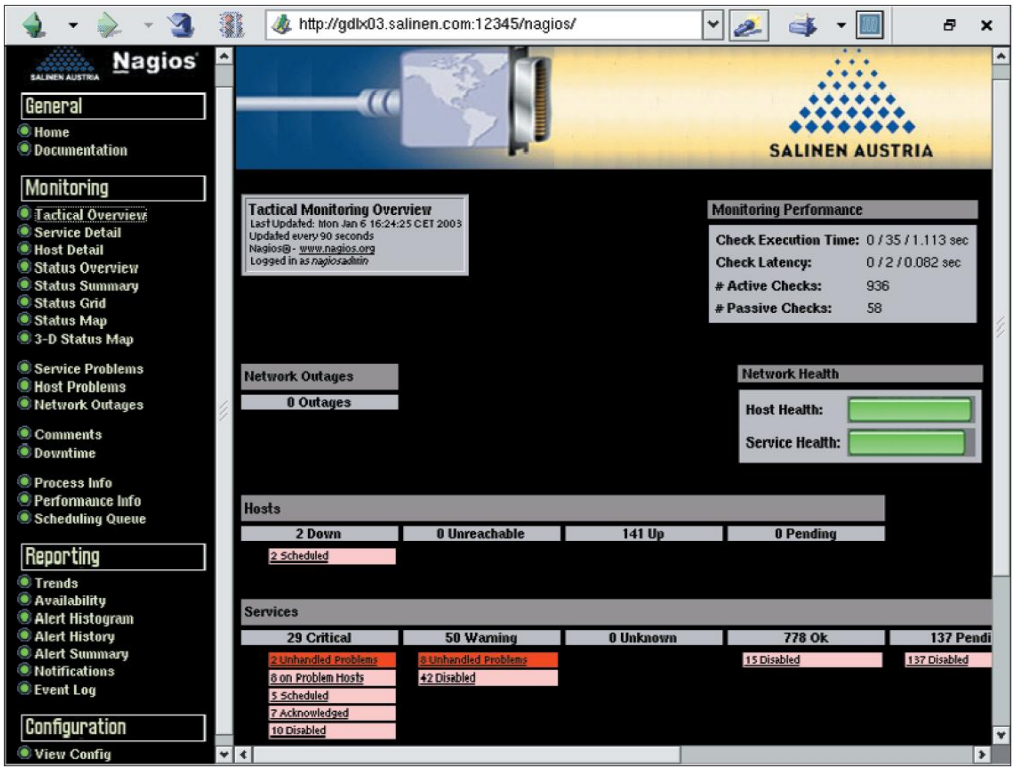


Figura 23 – Nagios *Tactical Monitoring Overview* (baseado em [Ruzicka, 2003])

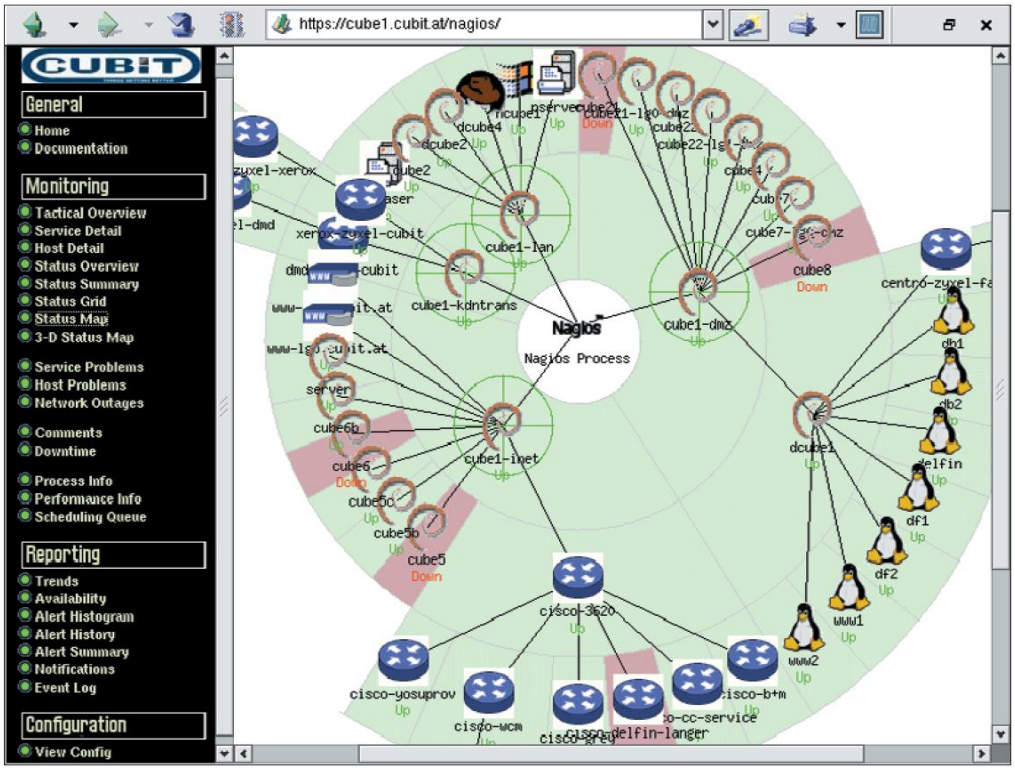


Figura 24 – Mapa de estados do Nagios (baseado em [Ruzicka, 2003])

O resumo de alertas é novo e ajuda o administrador a descobrir rapidamente servidores vulneráveis. O histograma de alertas é um outro recurso novo que mostra a acumulação dos problemas. Os administradores também podem manipular os processos do Nagios através da interface Web usando-a para ativar ou desativar testes individuais, adicionar tempos de inatividade, comentários ou reiniciar processos [Ruzicka, 2003].

3.4.7 Alertas na rede

Em caso de existirem alertas, normalmente o administrador prefere ser notificado em vez de abrir a interface Web para descobrir o que aconteceu. O Nagios reflete isso gerando notificações sempre que um '*hard state*' muda. Filtros especiais evitam que o administrador seja bombardeado com milhares de mensagens. O primeiro nível de filtro permite que sejam ativas ou desativas globalmente as notificações. O segundo estágio fornece filtros para serviços e *hosts* dividido em quatro níveis diferentes. Para além disso o Nagios não vai notificar o administrador quando:

- O tempo de inatividade está previsto para o *host* ou serviço;
- Um componente está a oscilar entre dois estados (*flapping*);
- As notificações foram desativas globalmente para um componente, ou;
- O problema ocorreu fora do período de notificação.

O último nível corresponde a filtros de contatos. O administrador pode definir o estado que o Nagios terá de atingir antes de notificar cada utilizador, que pode ser '*Warning*' ou '*Critical*'. Estas definições podem ser baseadas nos *hosts* e/ou serviços. Este nível também permite o administrador definir notificações temporais para o utilizador [Ruzicka, 2003].

3.4.8 Contactos

O Nagios usa grupos de contactos para encontrar qual o administrador que deve ser notificado para um grupo específico de *hosts* ou serviços e previne que um administrador receba mensagens idênticas mais do que uma vez. Não existem restrições de vias usadas para as mensagens, *email*, SMS ou serviços de mensagens instantâneas, como o ICQ são opções típicas. O Nagios contém vários níveis de alertas (*notification escalation*), permitindo uma notificação dos processos sem restrições. Normalmente e sem escalonamento, no caso de haver algum problema, o Nagios envia um alerta único e repete a mensagem após um intervalo definido. O administrador do sistema pode definir as mensagens e os intervalos para os casos mais urgentes (ver Figura 25)[Ruzicka, 2003].

```
01 # generic contact definition
02 define contact{
03    name                            generic-contact
04    service_notification_period     24x7
05    host_notification_period        24x7
06    service_notification_options    w,c,r #warning,
critical, recovery
07    host_notification_options       d,r #down, recovery
08    service_notification_commands   notify-by-email
09    host_notification_commands      host-notify-by-email
10    register                        0
11 }
12
13 # 'nagios' contact definition
14 define contact{
15    use                             generic-contact
16    contact_name                    nagios
17    alias                           Nagios Admin
18    email                           nagios
19 }
```

Figura 25 – Exemplo de uma definição de contactos (baseado em [Ruzicka, 2003])

A maior parte das vezes a configuração do Nagios é efetuada através de ficheiros de texto, que normalmente são editados à 'mão' através de editores. Uma alternativa a este processo é usar uma ferramenta chamada NagiosQL [nagiosQL, 2012] onde é possível, de uma forma gráfica, fazer todas as configurações de *hosts*, *services*, *contacts*, *hostgroups*, etc. O NagiosQL é uma ferramenta profissional baseada em Web que necessita de um servidor que inclua suporte a PHP e a base de dados (ver Figura 26).

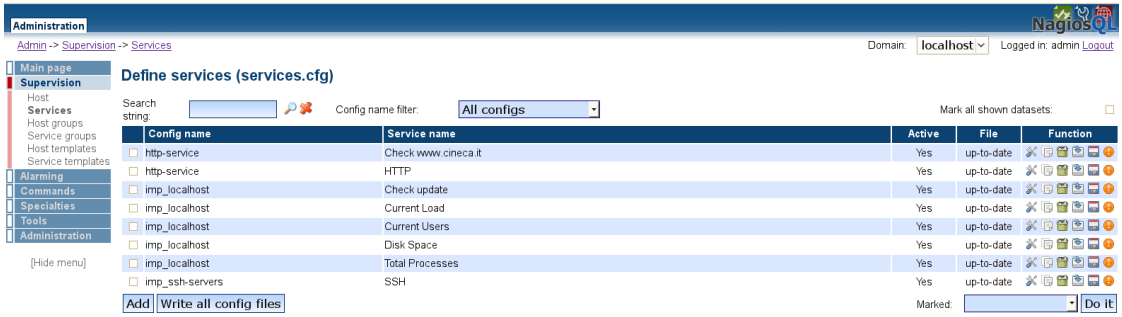


Figura 26 – Interface do NagiosQL

Passando vários anos desde o Netsaint, o Nagios é agora um produto que provou ser uma solução de topo para monitorização de rede em código aberto. Compete contra a maioria das aplicações comerciais, e na maioria dos casos tem um menor custo de implementação e um maior nível de eficácia do que muitas soluções comerciais. Tornou-se uma aplicação que é flexível e de manutenção relativamente fácil. Para cada problema que surge existe uma forma de monitoriza-lo através do Nagios usando os *plug-ins* [Plugins, 2011] da comunidade ou implementando um de forma a corresponder a todas as necessidade que o Nagios contém [Max Schubert, 2008].

3.5 Port Mirror, Netflow e sFlow

Softwares como o Nagios ou MRTG coletam informações através dos protocolos SNMP e ICMP e mediante os dados recolhidos podem gerar alarmes e produzir relatórios. No entanto, estes *softwares* não fornecem muitos pormenores sobre os dados que circulam na rede. Quando se pretende ter mais detalhes sobre esses dados podem ser utilizadas tecnologias como o Port Mirror, Netflow ou sFlow.

3.5.1 Port Mirror

Um analisador de protocolos captura todo o tráfego que passa numa rede, podendo fazê-lo ao nível do *byte*. Existem *softwares* para fazer este trabalho como por exemplo o Wireshark [wireshark, 2011], com estes *softwares* é possível capturar todos os pacotes e descodificá-los de acordo com o protocolo e assim saber que tipo de tráfego circula pela rede desde a *frame Ethernet* até ao protocolo de aplicação. Para que estas ferramentas consigam fazer a captura do tráfego este tem que ir até à máquina onde o *software* está instalado. Esta particularidade não é problema para uma rede que utilize *Hubs*, mas já o é para uma rede que utilize *switchs*, pois nem em todos os interfaces do *switch* passa todo o tráfego, mas sim apenas o que realmente se destina a ele. A aplicação desta técnica exige que a placa de rede da máquina de captura esteja configurada no modo promíscuo, para que esta rececione todos os pacotes do segmento da rede e não apenas os que lhe estão endereçados.

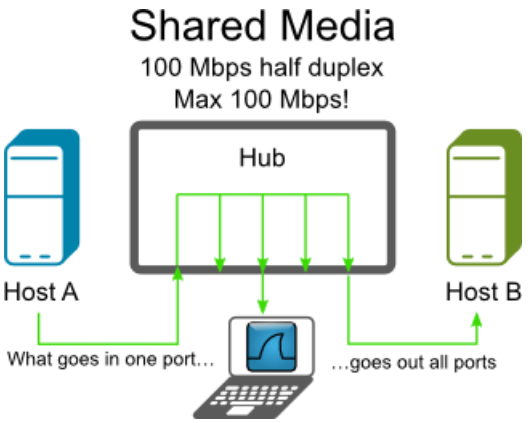


Figura 27 – Captura usando *HUB* (baseado em [Wireshark, 2011])

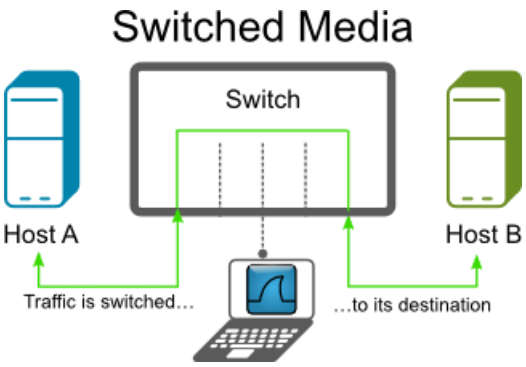


Figura 28 – Captura usando um *switch* (baseado em [Wireshark, 2011])

Nas Figura 27 e Figura 28 estão ilustrados os conceitos apresentados. Na Figura 27 o tráfego que vai do *Host A* para o *Host B* (Porta 1 para a Porta 5) é replicado em todas as portas, sendo assim o *sniffer* colocado na porta 3 irá receber todo o tráfego. Na Figura 28 o *sniffer* colocado na mesma porta não receberá o tráfego destinado ao *Host B* já que neste caso está a ser usado um *switch* e este encaminha o tráfego apenas para a porta respetiva. Uma

técnica por vezes usada quando realmente se quer capturar o tráfego de um dos portos do *switch*, é colocar um *HUB* na porta onde é pretendida a análise do tráfego. Esta técnica tem algumas desvantagens que se prendem na necessidade de haver uma deslocação ao local onde se encontra o *switch* e haver dificuldades na troca dos portos. No caso de redes *Gigabit* esta técnica também não é uma boa solução já que não existem *HUBs* para estas velocidades (ver Figura 29).

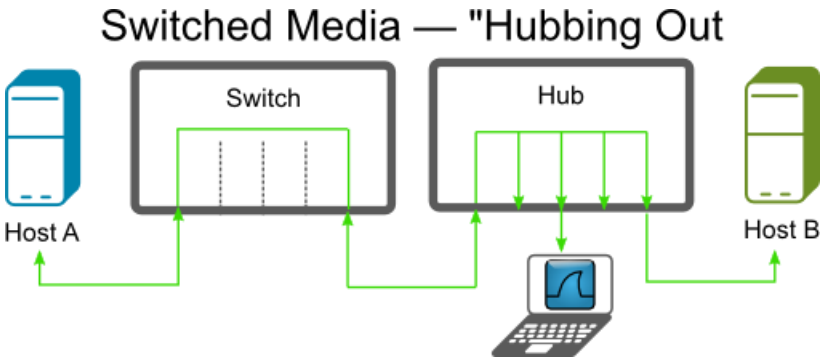


Figura 29 – Captura usando um *HUB Ethernet* (baseado em [Wireshark, 2011])

Port-Mirror é uma função que alguns *switchs* têm que facilita o uso dos analisadores de protocolos. Com o *Port-Mirror* é possível 'espelhar' o tráfego de uma ou mais portas, (portas estas que são as que se pretendem monitorizar) para outra porta, onde se encontra um analisador de protocolos.

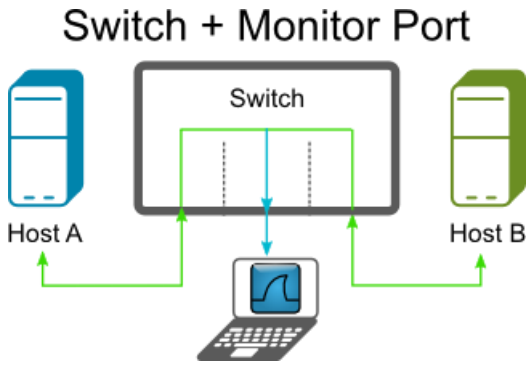


Figura 30 – Captura usando *switch* com *Port Mirror* (baseado em [Wireshark, 2011])

No exemplo da Figura 30 a porta 5 onde está instalado o *Host B* está a ser copiada para a porta 3 onde está o analisador de protocolos e assim é possível capturar os pacotes. A porta a monitorizar pode ser facilmente trocada enviando um comando para o *switch* não sendo assim necessário a troca de qualquer cabo. Outras funções podem ser selecionadas como por exemplo, indicar que se pretende capturar o tráfego apenas num sentido, por exemplo do *Host B* para o *Host A*. O *Port-Mirror* pode ser uma função muito útil na análise de protocolos mas por norma só está disponível em *switchs* mais dispendiosos [Wireshark, 2011].

3.5.2 NetFlow

O Netflow é um protocolo de rede desenvolvido e patenteado pela Cisco System em 1996 para recolher informação sobre tráfego IP de uma rede. Este protocolo tornou-se num *standard* da indústria para a monotorização de tráfego de rede sendo atualmente suportado por várias plataformas tais como Cisco IOS, NX-OS e por vários sistemas operativos como o Linux, FreeBSD, NetBSD e OpenBSD, sendo usado por vários fabricantes como por exemplo, Juniper, Enterasys Switchs, Alcatel e Foundry. O Netflow permite que dispositivos sejam eles *routers* ou *switchs* (*Probes*), tenham a capacidade de gerar registos que podem ser enviados a um *collector* através da rede e estes dados poderão ser analisados em tempo real, manualmente ou através de um *software* de análise denominado *Network Collector* (ver Figura 31) [Analyser, 2011].

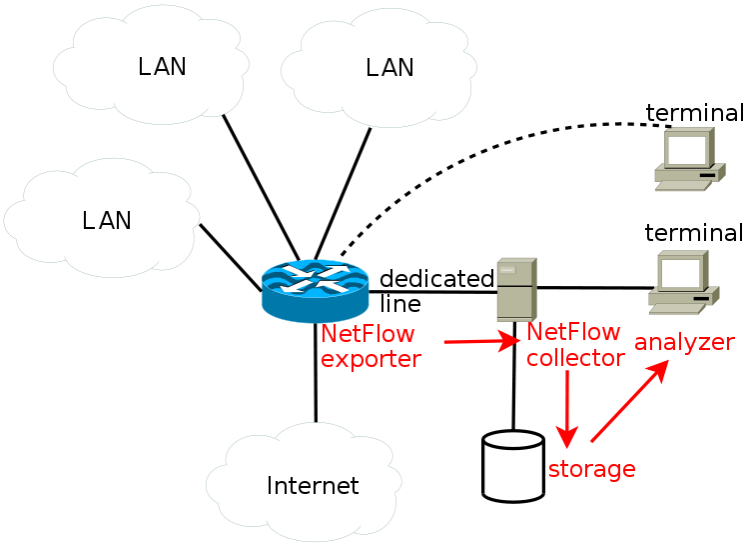


Figura 31 – Funcionamento do Netflow (baseada em [Netflow, 2012])

Para que se possa recolher informação sobre tráfego, uma interface de rede é configurada com o Netflow e quando um certo fluxo de dados é identificado nesse interface, o primeiro pacote e todos os restantes começam a ser contabilizados na tabela de fluxos registando o número de octetos e de pacotes. Os dados recolhidos serão mais tarde exportados através do protocolo UDP (*User Datagram Protocol*) ou SCTP (*Stream Control Transmission Protocol*) para um *collector* que tem uma porta configurada para esse fim. Os pacotes são constituídos por um cabeçalho e em seguida vem os fluxos respetivos para cada tipo de tráfego [Leobino Sampaio, 2002].

Na Figura 32 é possível visualizar uma tabela de fluxos guardada na *cache* de um dispositivo com suporte Netflow, estes registos permanecerão até que uma das condições ocorra:

- Término da conexão TCP através de TCP FIN ou RST;

- O tamanho máximo da tabela foi atingido;
- O tempo de inatividade expirou (Excesso de tempo sem pacotes com a característica do fluxo);
- O tempo de atividade expirou (Tempo máximo que um fluxo pode permanecer na tabela).

```
rs-bb3>sh ip cache flow
IP packet size distribution (8670M total packets):
  1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
  .001 .453 .058 .016 .010 .006 .006 .004 .005 .004 .003 .004 .004 .003 .003

   512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
  .003 .002 .064 .024 .319 .000 .000 .000 .000 .000 .000
IP Flow Switching Cache, 4456704 bytes  82 active, 65454 inactive, 4120535 added
   89798660 ager polls, 0 flow alloc failures  Active flows timeout in 1 minutes
   Inactive flows timeout in 15 seconds  last clearing of statistics never
Protocol         Total    Flows   Packets Bytes  Packets Active(Sec) Idle(Sec)------
   --          Flows     /Sec     /Flow  /Pkt     /Sec     /Flow     /FlowTCP-
   Telnet      834600      0.7        16   166     12.3       9.0      16.2TCP-
   FTP        5017218      4.4        10    86     46.1       4.4      15.7TCP-
   FTPD       2026874      1.8       278   992    504.1      45.1       2.7TCP-
   WWW      218687228    195.5        14   506   2791.2       5.9       9.3TCP-
   SMTP      27937130     24.9        10   385    268.9       6.1      11.4TCP-
   other    115224269    103.0        31   671   3253.9      12.9      12.0UDP-
   DNS       46205254     41.3         3    76    161.5       4.3      17.2UDP-
   NTP        2290487      2.0         1    75      2.5       0.2      17.3UDP-
   TFTP           794      0.0         3   212      0.0       2.2      16.3UDP-
   Frag         40824      0.0       154   734      5.6      49.8       4.7UDP-
   other    120392993    107.6         4   176    508.3       1.9      17.4ICMP
            19163422     17.1         5   101     93.2       4.0      17.1IGMP
               13612      0.0         1    28      0.0       0.0      15.5IPINIP
```

Figura 32 – Dados Netflow (baseado em [Andrey Vedana Andreoli, 2006])

Dos dados coletados de cada pacote é registado o seguinte: Endereço IP de origem, Endereço IP de destino, Porta de origem, Porta de destino, Protocolos, Tipo de serviço, Interface de origem do tráfego.

3.5.3 sFlow

O sFlow é um protocolo com algumas características idênticas ao Netflow mas no geral é mais simplificado. Também funciona numa estrutura *Probe* (sonda) e *Collector*, mas uma das principais diferenças é que a sonda não faz a coleta de todo o tráfego, mas apenas de algumas amostras, esta taxa de amostragem pode ser configurada, no entanto o valor usual é de 1 em 100 pacotes. Uma das vantagens do sFlow relativamente ao Netflow é que gera muito menos tráfego na rede e após um prazo razoável de coleta informa sobre quais as tendências da rede [sFlow.org, 2003].

O sFlow é uma tecnologia de amostragem multi - fabricante incorporada em *switchs* e *routers*. Ela fornece a capacidade de monitorizar de uma forma continua os fluxos de tráfego a altas velocidades em todos os interfaces em simultâneo. Ao fornecer visibilidade sobre o uso da rede e das rotas ativas em redes complexas e de altas velocidades, o sFlow fornece os dados necessários para controlar e monitorizar o uso da rede de uma forma eficiente,

garantindo que os serviços da rede fornecem uma vantagem competitiva. Alguns exemplos de aplicações do sFlow são:

- Detetar, diagnosticar e corrigir problemas na rede;
- Gerir congestionamento em tempo real;
- Entender aplicações do tipo P2P, Web, DNS;
- Analisar a rede para Identificar atividades não autorizadas e rastrear as fontes dos ataques do tipo '*denial-of-service*';
- Planear a capacidade [sFlow.org, 2003].

Num ambiente comutado o local mais eficiente para fazer a monitorização de tráfego é dentro de um *switch* ou de um *router*, onde todo o tráfego é visto. Sondas tradicionais só têm uma visão parcial do tráfego. No entanto, uma solução incorporada dentro de um *switch* ou *router* não deve afetar o desempenho de encaminhamento. Os *switches* e *routers* com a tecnologia de amostragem sFlow estão disponíveis desde 2001. Esta solução fornece medições de tráfego detalhadas, a velocidades Gigabit, permitindo obter uma visão sobre as decisões de encaminhamento, não constituindo um problema no encaminhamento da rede e do desempenho da mesma [sFlow.org, 2003].

O sFlow é uma tecnologia de amostragem de pacotes que atende aos requisitos fundamentais para uma solução de monitorização do tráfego de rede. A amostragem de pacotes tem sido utilizada para monitorizar o tráfego de rede à mais de 20 anos (ver Figura 33). A empresa Hewlett-Packard fez a primeira demonstração de monitoração de uma rede usando amostragem de pacotes na universidade de Geneva no CERN em 1991. Seguiu-se com a introdução de produtos de rede com capacidade de amostragem de pacotes incluída HP Extendend RMON - em 1993. [sFlow.org, 2003].

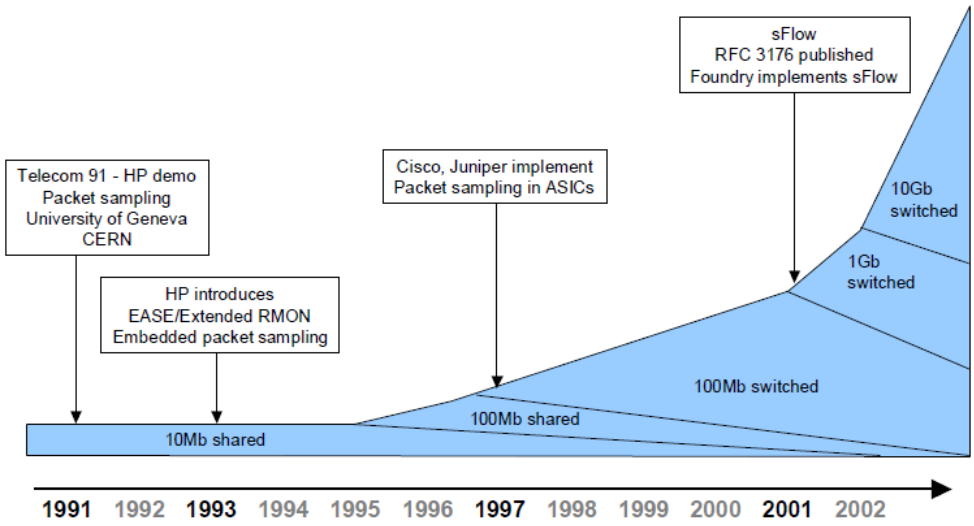


Figura 33 – História sobre a amostragem de pacotes (baseado em [sFlow.org, 2003])

O sFlow oferece uma visão do uso da rede e das rotas ativas. É uma técnica escalável para a medição do tráfego da rede, coleta, armazena e analisa os dados do tráfego permitindo que dezenas de milhares de interfaces sejam monitorizadas a partir de um único local. O sFlow é escalável e permite monitorizar ligações com velocidades até 10Gb/s sem afetar o desempenho dos *routers* e dos *switchs* adicionando uma carga na rede menos significativa que outros métodos, por exemplo o Netflow. É uma solução de baixo custo, implementada numa ampla gama de dispositivos, desde simples *switchs* L2 a *routers*, sem a necessidade de CPU e memória adicionais [sFlow.org, 2003].

O agente sFlow é um processo de *software* que é executado como parte do *software* de monitorização de rede dentro de um dispositivo (ver Figura 34). Ele combina contadores de interface e amostras de fluxo em data-gramas sFlow que são enviados através da rede para o coletor sFlow. A amostragem dos pacotes é normalmente realizada pelos ASICs de comutação/roteamento, proporcionando alto desempenho. O estado da tabela de encaminhamento/roteamento associado a cada pacote, também é gravado [sFlow.org, 2003].

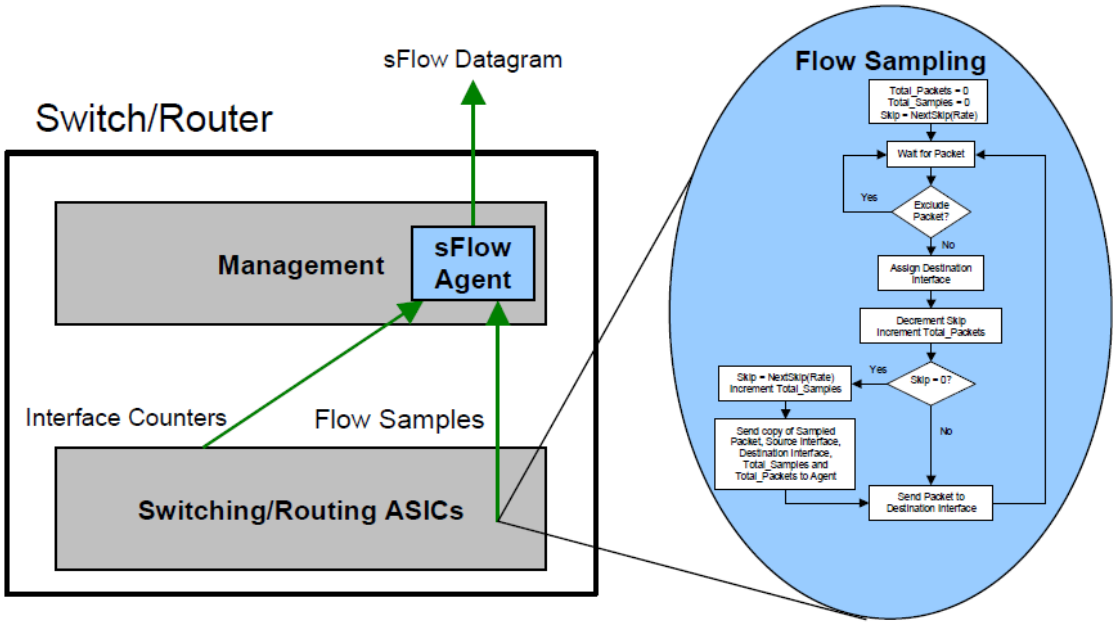


Figura 34 – Agente sFlow incorporado num *switch/router* (baseado em [sFlow.org, 2003])

O agente SFlow tem muito pouco processamento, simplesmente insere os dados em data-gramas sFlow e imediatamente os envia para a rede. O envio imediato dos dados minimiza os requisitos de memória e CPU associados ao agente sFlow. A Figura 35 mostra os elementos básicos de um sistema. Os agentes enviam de forma contínua um fluxo de data-gramas sFlow para a central sFlow *Collector* onde eles são analisados para produzir uma vista detalhada e em tempo real dos fluxos de tráfico de toda a rede [sFlow.org, 2003].

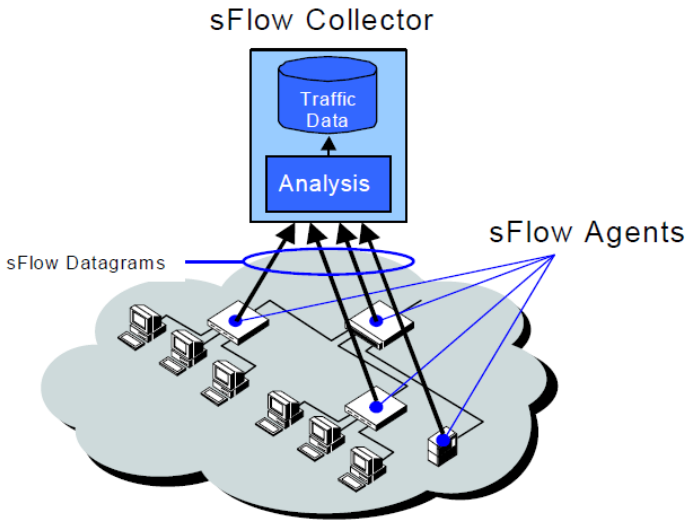


Figura 35 – Agente e Coletor sFlow (baseado em [sFlow.org, 2003])

3.6 Ntop

O Ntop [Ntop, 2011] (*Network Traffic Probe*), é um *software Open Source* desenvolvido na universidade de Pisa em Itália e está disponível para as plataformas Windows e Unix [unix, 2012]. Permite verificar quais os sistemas de uma rede que estão ativos e quais os tipos de tráfego enviado e recebido desses sistemas. O tráfego de rede é classificado por protocolo e alguns dos mais comuns são reconhecidos, incluindo os tráfegos TCP/UDP e os seus associados. Esta ferramenta também suporta protocolos que não sejam IP, como o Decnet, IPX, AppleTalk e outros. A utilização destes protocolos por parte do Ntop é configurável podendo através destes ser gerados relatórios de informações sobre IP (*Internet Protocol*) e FC (*Fiber Chanel*). A lista abaixo exibe os protocolos passiveis de ser monitorizados pelo Ntop:

- TCP/UDP/ICMP
- (R)ARP
- IPX
- DLC
- IPsec
- AppleTalk
- NetbiosTCP/UDP
    - FTP
    - HTTP
    - DNS
    - Telnet
    - SMTP/POP/IMAP
    - SNMP
    - NFS
    - X11

- Fibre Channel
    - Control Traffic - SW2,GS3,ELS
    - SCSI

Esta ferramenta difere das ferramentas '*Packet Sniffer*' porque tem como objetivo fornecer dados estatísticos sobre os pacotes da rede, e não sobre os seus conteúdos. Os resultados podem ser visualizados de uma forma gráfica através do *browser*, quer seja utilizado o próprio servidor Web, quer seja utilizado outro como por exemplo o Apache.

A vantagem em utilizar o Ntop é que este disponibiliza uma forma rápida e precisa de obter informações da rede sem recorrer ao uso de uma sonda (*network probe*) ou de um dispositivo *Sniffer*. Em muitos casos é necessário uma sonda dedicada para rastrear problemas na rede, mas às vezes o tempo é crítico e nessas situações o Ntop pode ser a solução adequada.

Paralelamente convém realçar que em determinadas configurações de rede pode não ser possível ligar uma sonda, como por exemplo quando os sistemas estão interligados via WAN. Por último, pode-se referir a questão dos custos associados, comprar sondas é mais dispendioso do que instalar um *software* livre [Maxwell, 1999].

No entanto, o Ntop também possui desvantagens, nomeadamente porque pode suscitar problemas aos sistemas. O facto de ser necessário colocar o interface de rede em modo promíscuo, implica que todo o tráfego vá ser capturado, resultando num processamento significativo no sistema, pelo que deve haver um cuidado adicional na sua execução. A criação de alguns mecanismos automatizados pode ser uma solução, uma vez que o Ntop só será executado quando realmente existir essa necessidade e nunca durante momentos críticos, ou em picos de utilização do sistema. O uso do Ntop também levanta questões ao nível de segurança, dado que o acesso não autorizado a esta ferramenta pode resultar numa exposição adicional de vulnerabilidades existentes, dados, sistemas e falhas de segurança [Maxwell, 1999].

A informação estatística obtida pelo Ntop encontra-se num conjunto de relatórios em páginas HTML e pode ser consultada através de um *Web Browser* (normalmente na porta 3000) em vários locais ao mesmo tempo. Os relatórios que podem ser visualizados são vários incluindo:

- Data Sent + Received;
- Traffic Statistics Report;
- Host Information Report.

Alguns dos mais importantes relatórios serão descritos com mais detalhe em seguida.

3.6.1 Data Sent + Received

Este relatório mostra o tráfego de rede que foi recebido e enviado pelos *hosts* ativos. A tabela pode vir organizada por ordem crescente ou decrescente de um dos parâmetros, sendo estes a quantidade de dados recebidos e enviados em *bytes*, a quantidade de dados do sistema em valores percentuais e a quantidade de dados dividida por protocolos usados. A apresentação dos dados pode ser customizada, podendo apresentar os dados de toda rede, apenas das máquinas locais, ou apenas das máquinas remotas, também é possível apresentar os dados enviados e recebidos de forma individual [Maxwell, 1999].

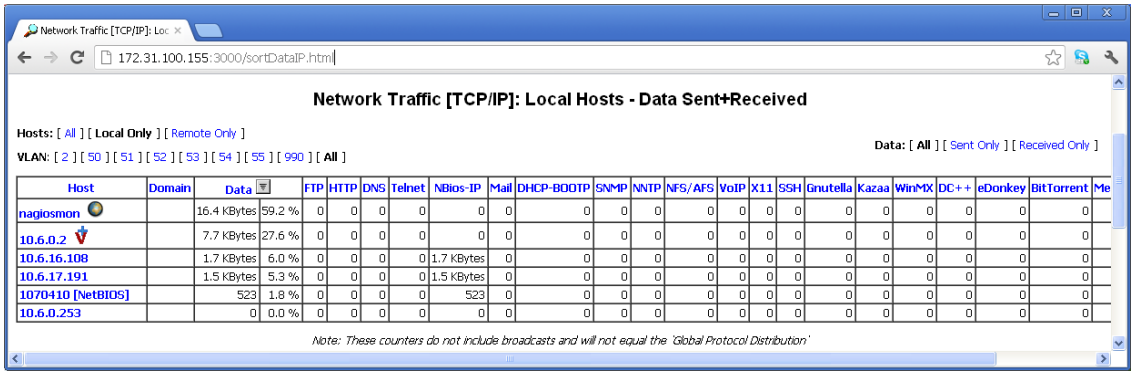


Figura 36 – Relatório Ntop de dados enviados e recebidos

No exemplo ilustrado na Figura 36 algumas máquinas foram identificadas: nagiosmon, 10.6.0.2, 10.6.16.108 entre outras, sendo estes os sistemas que presentemente estão a gerar tráfego na rede. Através deste relatório também é possível visualizar que a máquina nagiosmon é responsável por 59.2 % do tráfego.

3.6.2 Traffic Statistics Report

Este relatório mostra dados estatísticos globais sobre o tráfego e a sua distribuição por protocolo e portas. Os valores podem ser analisados através de um conjunto de tabelas e gráficos. Existem várias variáveis interessantes de analisar: *Total Traffic* quantifica o total de dados que foram lidos a partir do interface sendo este valor a soma do tráfego IP e não IP. O campo *Total Packets Processed* representa o número total de *frames* capturados desde que a monitorização ao interface Ntop teve início (ver Figura 37).

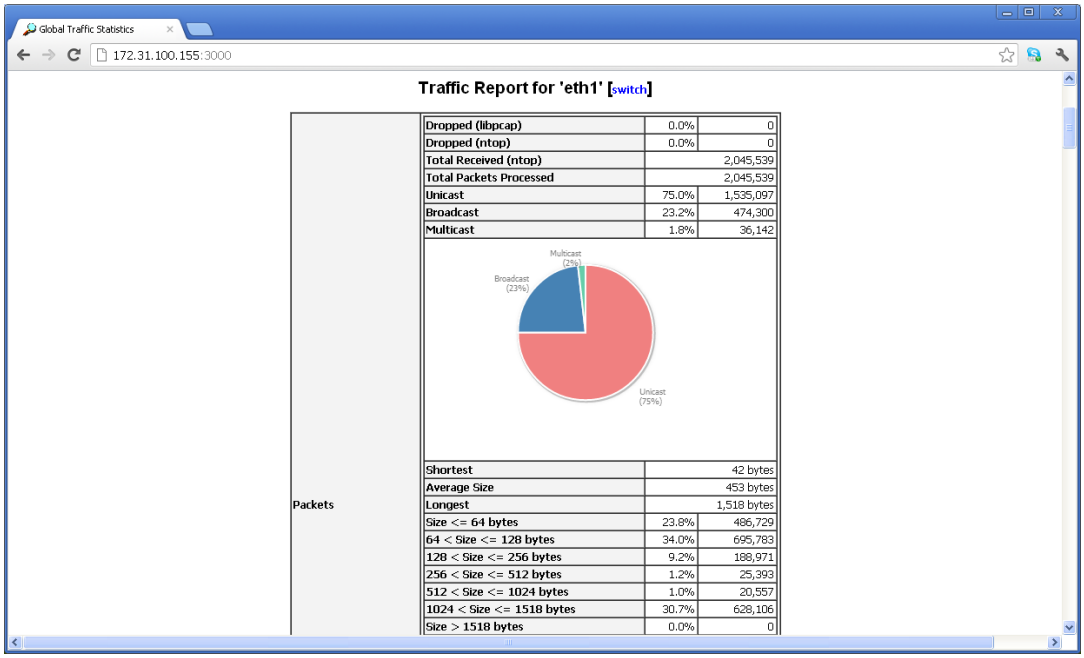


Figura 37 – Relatório Ntop com estatísticas sobre o tráfego

3.6.3 Host Information Report

Este relatório contém informações adicionais sobre as máquinas encontradas na rede. A identificação da máquina pode ser feita através do seu *hostname*, mas caso este não possa ser mapeado será usado o seu IP. Os dados são apresentados a partir de um conjunto de tabelas e gráficos podendo-se visualizar informações sobre estatísticas temporais de tráfego, estatísticas sobre pacotes, protocolos, portas e serviços usados e sessões TCP/UDP ativas.

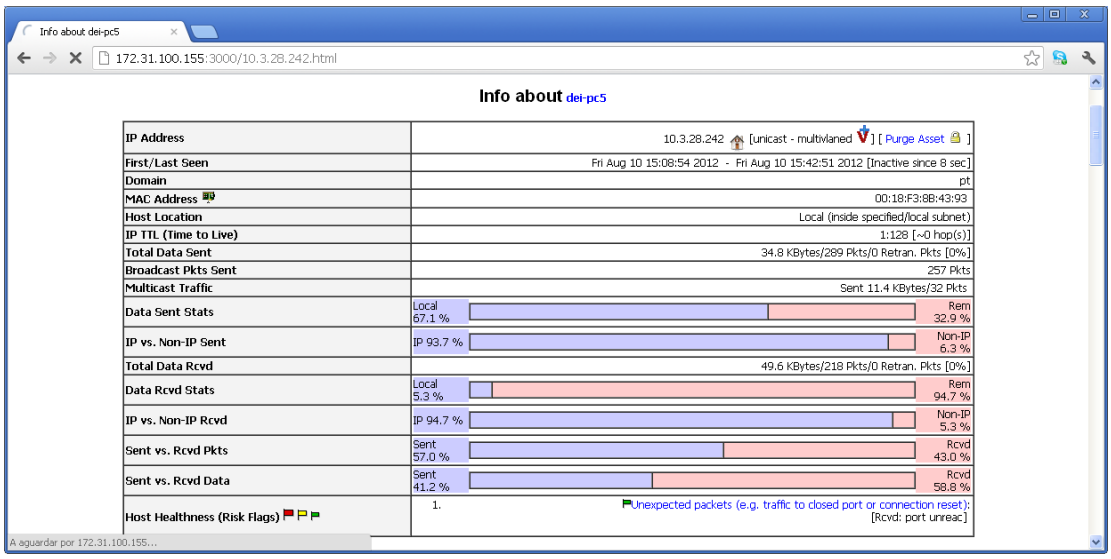


Figura 38 – Relatório Ntop com estatísticas sobre as máquinas da rede

De seguida apresentam-se as principais funções executadas pelo Ntop:

- Classificação do tráfego da rede de acordo com vários protocolos;
- Visualização de relatórios do tráfego de rede, de acordo com vários critérios;
- Visualização de estatísticas do tráfego;
- Armazenamento no disco de estatísticas de tráfego em formato RRD;
- Identificação das identidades dos utilizadores (Ex. endereços de *email*)
- Identificação dos sistemas operativos dos utilizadores;
- Criação de relatórios de Tráfego IP de acordo com vários protocolos;
- Análise de tráfego IP e classificação de acordo com a sua origem/destino;
- Visualização da Rede IP (quem comunica com quem);
- Criação de relatórios de uso de IP classificado por tipo de protocolo;
- Funcionamento como um coletor NetFlow / sFlow para fluxos gerados por *routers* ou *switchs*;
- Produção de estatísticas de tráfego RMON;
- Permite efetuar consultas às informações através de um browser, pelo facto de possuir um *Web Server* integrado [ntop, 2012].

3.7 Vulnerabilidades em Redes Locais

3.7.1 Confidencialidade

Numa rede de computadores existem inúmeras interações. A confidencialidade destas pode subdividir-se em dois grupos: confidencialidade de tráfego e confidencialidade de conteúdos. Sendo que a primeira esconde os interlocutores finais da interação e a segunda os dados úteis trocados pelos mesmos. As primeiras versões dos protocolos IP não possuíam nenhuns mecanismos para que os dados que circulavam na rede fossem ilegíveis a terceiros não autorizados, pelo que qualquer máquina com acesso à rede poderia obter dados, nomeadamente relativos a tráfego e conteúdos. A obtenção de dados pode ser efetuada mesmo quando não existe acesso físico a determinadas partes da rede, usando nestes casos técnicas de redireccionamento como ARP *poisoning* ou DNS cache *poisoning*, assim será possível fazer passar as mensagens pela máquina ou rede do atacante [Zúquete, 2006].

A certa altura a falta de mecanismo de suporte à confidencialidade tornou-se um problema de segurança grave, fazendo com que na década de 1990 surgissem diversas soluções ao nível de toda a pilha protocolar. A confidencialidade dos conteúdos é garantida cifrando-os limitando assim as entidades que os conseguem compreender (ver Tabela 6) [Zúquete, 2006].

<table>
<tr><th>Nível Protocolar</th><th colspan="2">Soluções</th></tr>
<tr><td>Transação</td><td colspan="2">PGP, PEM, S/MIME, SHTTP, SET</td></tr>
<tr><td>Aplicação</td><td rowspan="3">SSH</td><td>HTTPS, IMAPS, POPS, LDAPS, NNTPS</td></tr>
<tr><td>Apresentação</td><td rowspan="2">TLS / SSL</td></tr>
<tr><td>Sessão</td></tr>
<tr><td>Transporte</td><td colspan="2"></td></tr>
<tr><td>Rede</td><td colspan="2">IPSec</td></tr>
<tr><td>Lógico</td><td colspan="2">GSM, WEP, DECT, Bluetooth</td></tr>
<tr><td>Físico</td><td colspan="2"></td></tr>
</table>

Tabela 6 — Protocolos de suporte à autenticação confidencial e autenticada (baseada [Zúquete, 2006])

Existem várias ferramentas com as quais é possível inspecionar tráfego e conteúdos sendo para fins administrativos ou maliciosos. Uma das mais conhecidas é o *tcpdump* para sistemas *unix*, sendo possível capturar e apresentar todos os pacotes da rede. Outra ferramenta é o W*ireshark* que permite uma melhor análise e uma filtragem mais fina dos protocolos da camada de aplicação [Zúquete, 2006].

A inspeção de tráfegos e conteúdos da rede pode ser completamente escondida e indetetável sendo esta uma característica fundamental para a camuflagem de sistemas de defesa NIDS. Existem vários dados importantes que circulando de uma forma aberta na rede podem ser causadores de riscos diversos, mas a circulação de *passwords* são das mais relevantes [Zúquete, 2006].

Captura de *passwords*

A autenticação através de memorização de *password* é uma forma de identificação de pessoas existente à mais tempo que os computadores. Neste processo existe um segredo partilhado a *password* entre quem faz a autenticação e o autenticador. Em sistemas computacionais existe a necessidade que o processo de transporte da *password* desde quem faz a autenticação até ao autenticador seja feito de uma forma segura, ou seja, não possa ser sujeito a escuta [Zúquete, 2006].

No início da *Internet* todas as aplicações distribuídas desenhadas usavam uma autenticação em que as *passwords* circulavam em 'claro' na rede. São exemplos: a iniciação de uma sessão remota com *telnet*, a transferência de ficheiros com FTP, o acesso a caixas de correio eletrónico usando POP e IMAP, o acesso a páginas protegidas via HTTP, entre outros. Nos dias de hoje estas aplicações ainda são usadas, no entanto, devem ser evitadas e recomenda-se a utilização de outras que utilizam protocolos mais seguros [Zúquete, 2006].

3.7.2 Prestação de Serviços

Os ataques à prestação de serviços (DoS – *Denial of Service Attacks*) podem ter várias motivações mas os objetivos são sempre os mesmos, fazer com que uma empresa ou serviço não consiga responder aos pedidos dos seus clientes. Para isso, sobre os sistemas operativos

ou servidores, podem ser usadas várias estratégias de ataque, mas existem três que são as mais usuais. A primeira consiste na exploração de vulnerabilidades conhecidas, por exemplo, o ataque *Ping-of-Death*. A segunda estratégia baseia-se na inundação de servidores com pedidos falsos mas bem formados, por exemplo, o ataque SYN *flooding attack*, estes pedidos sobrecarregam as máquinas alvo o que baixa a capacidade de reposta aos pedidos legítimos. Por último, a terceira forma de ataque consiste em inundar a rede de acesso à máquina servidora com tráfego inútil [Zúquete, 2006].

Ping of death (ICMP flood)

O utilitário *ping* usa o protocolo ICMP para testar a conectividade ao nível IP entre um, ou mais equipamentos. Este comando está integrado em vários sistemas operativos havendo apenas pequenas diferenças na sua implementação. Em sistemas Linux para efetuar o teste são enviados pacotes de 64 Bytes para o equipamento destino e são aguardadas respostas para cada um deles. No entanto, existe uma variação específica deste comando que pode causar a falha de um sistema, esta variante é denominada de *Ping of death* (Ping da morte) e a sua diferença para a versão original está no tamanho do pacote IP.

O RFC791 especifica que o tamanho máximo de um pacote IP é de 65535 Bytes. Sendo assim, uma solicitação de eco ICMP com mais de 65507 (65535-20-8) bytes de dados é ilegal mas passível de ser enviada devido à forma que o pacote é fragmentado. Tendo em conta estes factos, o Ping *of death* envia um pacote IP com mais de 65535 Bytes que ao serem remontados na outra extremidade num pacote completo fará transbordar o *buffer* de memória podendo causar danos ao sistema vítima (ver Figura 39) [SonicWALL, 2009] [Zúquete, 2006] [Maxwell, 1999] [Antoniou, S., 2009].

Figura 39 – Ataque *Ping of Death* (baseada [Antoniou, S., 2009])

SYN flooding

Estes ataques acontecem durante o *handshake* (aperto de mão a 3 tempos) que inicia uma sessão de comunicação entre duas aplicações. Em circunstâncias normais, a aplicação que inicia uma sessão envia um pacote de sincronização (SYN - *synchronize*). O recetor envia o pacote de reconhecimento (SYN-ACK), para indicar que recebeu o pedido, e então o emissor

envia um pacote de reconhecimento (ACK - *acknowledgment*). Após este *handshake* as aplicações estão configuradas para enviar e receber dados (ver Figura 40).

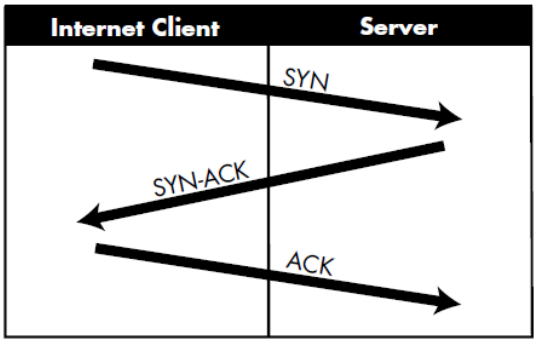


Figura 40 – Ligação normal entre o cliente e o servidor (baseado em [Zúquete, 2006])

Durante um ataque SYN Flood, o sistema alvo é inundado com uma série de pacotes SYN, sem enviar um ACK após as respostas do alvo. Por norma os segmentos SYN utilizam endereços IP falsos e em constante mudança, não sendo assim possível identificar o emissor. O servidor vai ficar algum tempo com uma série de meias ligações TCP, que só desaparecem quando ultrapassar o tempo máximo de espera pelo segmento ACK. Assim o servidor estará a ocupar o seu processador e a rede no envio de segmentos SYN-ACK inúteis. Desta forma o porto TCP pode ficar inacessível a pedidos legítimos já que este tem um certo limite para o número de meias ligações (ver Figura 41) [SonicWALL, 2009] [Zúquete, 2006].

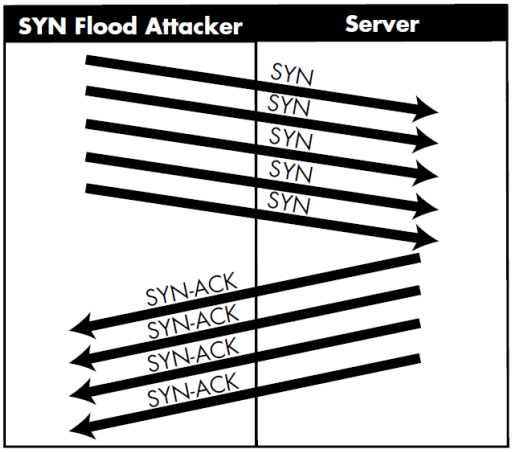


Figura 41 – Ataque SYN *flooding* (baseado em [Zúquete, 2006])

3.8 Análise

Neste capítulo foram abordadas duas ferramentas que podem ser usadas para a monitorização e apresentação gráfica do tráfego de uma rede, MRTG e Cacti. Após efetuado o estudo e implementação das duas ferramentas é possível efetuar uma análise comparativa.

O MRTG prima pela sua simplicidade, quer relativamente às opções disponíveis, quer na forma como são configuradas. No entanto, há uma característica que pode conduzir a uma menos correta leitura dos dados que deve ser salientada. Durante o processo de desenho do gráfico, caso exista uma falha na leitura dos dados, o MRTG continua utilizando o último valor recolhido o que pode conduzir a uma má apresentação dos dados.

Nesta situação o Cacti tem outra abordagem, já que não traça o gráfico durante o tempo onde a recolha dos dados não foi possível, não havendo assim dúvidas que não existem dados sobre aquele período. O Cacti tem uma diversidade de opções de configuração o que permite customização da aquisição de dados e da apresentação dos mesmos de diversas formas. Por vezes, esta possibilidade também constitui uma desvantagem já que tem dezenas de opções passíveis de ser configuradas, o que pode provocar algumas confusões.

A ferramenta Nagios é uma ferramenta de monitorização de redes de computadores que tem a capacidade de notificar o administrador sobre os eventos da rede. Ela tem uma configuração baseada em ficheiros de texto o que permite a reutilização de configurações anteriores, uma vez que a cópia é permitida. A utilização de ficheiros de texto também pode trazer desvantagens já que um pequeno erro de digitação pode invalidar o ficheiro de configuração. Para os utilizadores menos experientes neste tipo de configurações e que preferem configurações em ambientes gráficas existe a possibilidade da utilização de ferramenta nagiosQL, embora a sua instalação e configuração obriguem a algum trabalho.

Com o Ntop é possível fazer uma classificação do tráfego da rede e obter assim dados estatísticos sobre os pacotes. Uma das vantagens desta ferramenta é a rapidez com que é possível obter informações da rede. No entanto, o uso da mesma também levanta questões ao nível de segurança, dado que o acesso não autorizado a esta ferramenta pode resultar numa exposição adicional de vulnerabilidades existentes. O facto de ser necessário colocar o interface de rede em modo promíscuo também resulta num processamento significativo no sistema.

Outra forma de obter informações sobre o tráfego IP de uma rede é através dos protocolos Netflow e sflow. Com o Netflow é possível obter estatísticas bastante precisas. No entanto, o facto de não ser uma tecnologia multifabricante pode ser um problema. O sflow provoca menos carga na rede do que o Netflow, mas não coleta todo o tráfego, apenas uma amostragem o que pode não ser suficiente para situações mais específicas.

4 Caso de estudo – Rede DEI – ISEP

4.1 Introdução

Pretendia-se com este trabalho efetuar um estudo de vários protocolos / ferramentas de gestão e classificação de tráfego. Paralelamente pretendia-se instalar e configurar ferramentas de gestão e classificação de tráfego num cenário real que permitisse apresentar resultados práticos. O cenário escolhido foi a rede atual do Departamento de Engenharia do Instituto Superior de Engenharia do Porto (DEI/ISEP).

O projeto tem o nome **TrafficMonit** e na Figura 42 é possível visualizar o seu logo e os logos das ferramentas utilizadas para conseguir as funcionalidades pretendidas.

Para a monitorização do tráfego foi utilizado o Ntop, para gerar os alertas o Nagios e para apresentação dos dados o Cacti. Para que fosse possível que os alertas fossem transmitidos através de *email* o projeto usa a ferramenta Postfix [postfix, 2008]. Alguns dados são guardados em bases de dados RRD e outros em Mysql, para que fosse mais fácil consultar esses dados foi instalado o phpMyAdmin [phpmyadmin, 2008].

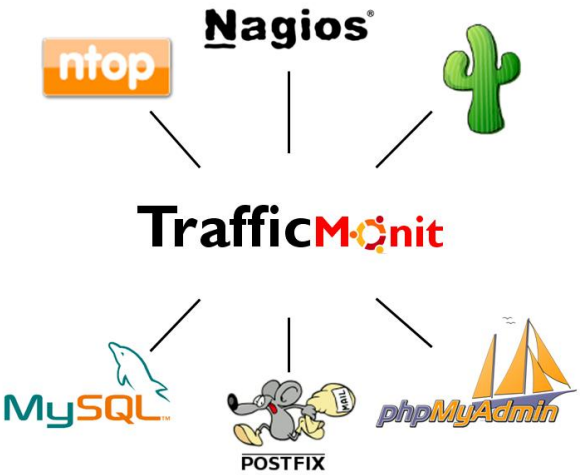


Figura 42 – Ferramentas do TrafficMonit

4.2 Monitorização Atual do DEI-ISEP

O DEI utiliza o MRTG para gerar gráficos de várias variáveis da rede de forma a poder gerir o seu desempenho. São gerados gráficos do *download* e *upload* de tráfego em routers, *switchs*, servidores, sendo eles físicos ou virtuais e pontos de acesso. Estes dados são recolhidos através do protocolo SNMP onde um gestor vai solicitando informações aos vários dispositivos da rede (agentes). Para além de estatísticas relativas a tráfego também são

monitorizados cargas dos CPU's, quantidade de processos, quantidade de utilizadores ligados por SSH entre outras contabilizações (ver Figura 43). Através do *software* Nagios é verificado se todos os componentes da sua infraestrutura estão ativos e também é testado o estado de vários serviços, tais como HTTP, FTP, *email*, verificação do nível dos tinteiros da fotocopiadoras, entre outros (ver Figura 44). Com estas ferramentas o DEI tem a possibilidade de mensurar o tráfego proveniente de vários pontos, mas não tem a possibilidade de identificá-lo e mensurá-lo por protocolo, ou saber qual a largura de banda que cada utilizador está a usar.

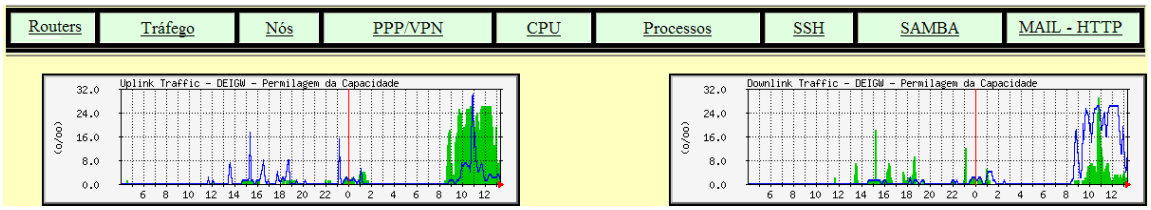


Figura 43 – Dados da rede DEI-ISEP recolhidos pelo MRTG (baseado em [DEI, 2012])

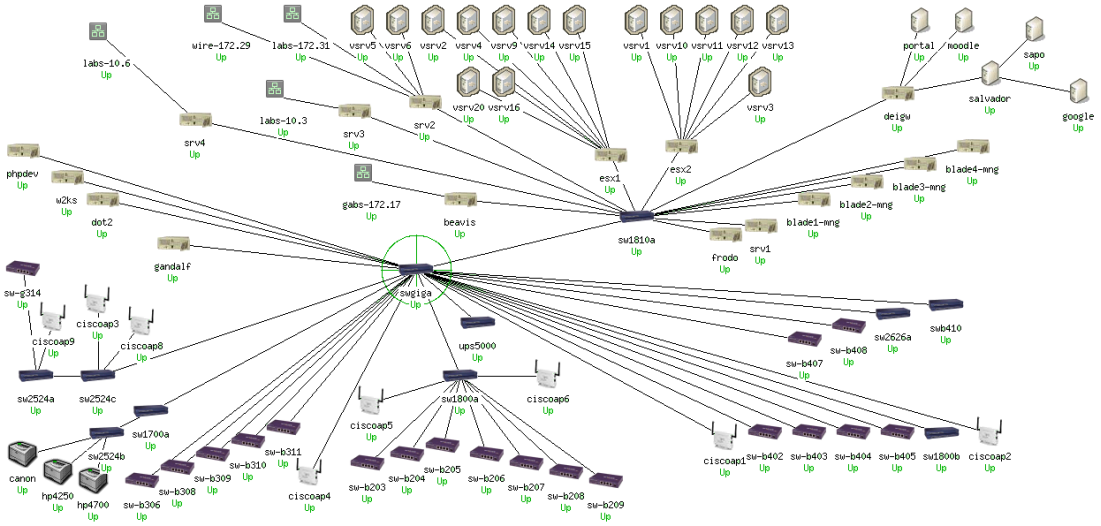


Figura 44 – Mapa Nagios da infraestrutura da rede do DEI-ISEP (baseado em [DEI, 2012])

Alguns dados sobre a monitorização da infraestrutura são acessíveis aos utilizadores da rede do DEI, mas os dados de administração não o são.

Para se compreender melhor a forma como funcionam estas ferramentas, no contexto deste cenário, foi implementado um cenário que replicasse o cenário real, onde também se configuraram as mesmas ferramentas, MRTG e Nagios.

Verificou-se que é possível diagnosticar vários tipos de problemas que surjam, tanto com os alertas do Nagios, como pela análise dos gráficos gerados pelo MRTG. No entanto, também

permitiu identificar uma função que não estava contemplada e que era o objeto de estudo desta tese, análise de tráfego por protocolo.

4.3 Implementação de Cenário com Ntop

A rede do DEI é essencial para o negócio do departamento pelo que optou-se por simular o cenário para efetuar o trabalho apresentado aqui nesta tese. Sendo assim, para testar o Ntop foi criada uma configuração típica onde o *software* foi instalado num computador com dois interfaces de rede em que um estava ligado à internet e outro à rede local. Para que a rede local tivesse acesso à internet foi configurado o DNS e foram feitas as configurações necessárias no NAT e no *firewall* (ver Figura 45).

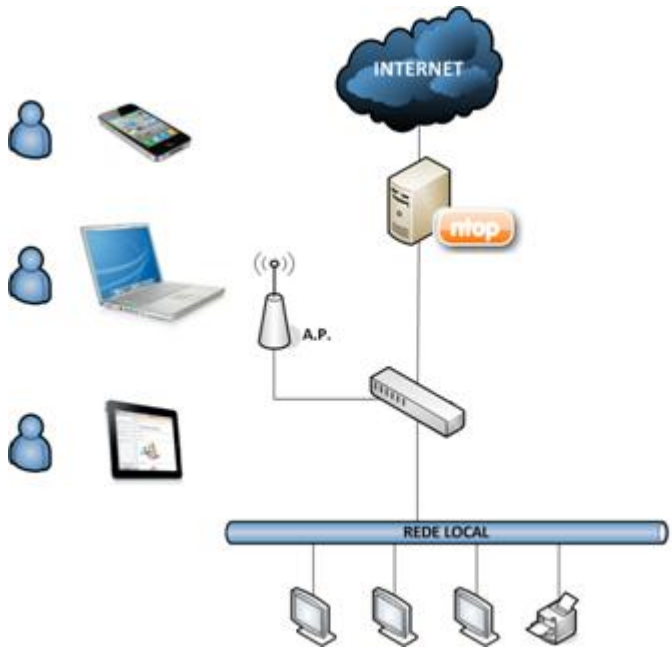
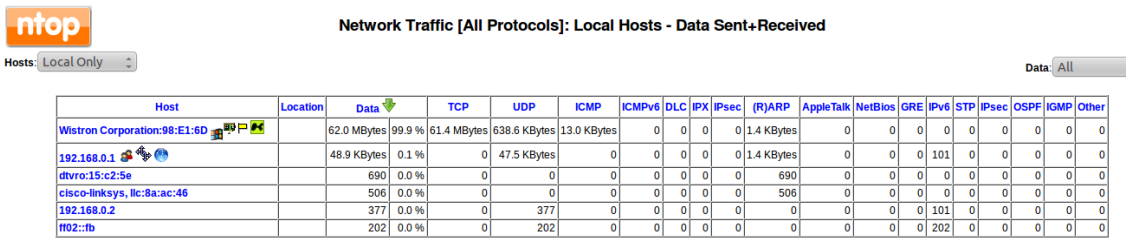


Figura 45 – Estrutura da implementação com Ntop

Após as várias configurações necessárias foi possível através do Ntop visualizar estatísticas sobre o tráfego, por exemplo, a Figura 46 apresenta através de uma tabela dados sobre o tráfego da rede interna, separados por protocolo.



ntop

Network Traffic [All Protocols]: Local Hosts - Data Sent+Received

Hosts: Local Only

Data: All

Host	Location	Data	TCP	UDP	ICMP	ICMPv6	DLC	IPX	IPsec	(R)ARP	AppleTalk	NetBios	GRE	IPv6	STP	IPsec	OSPF	IGMP	Other
Wistron Corporation:98:E1:6D		62.0 MBytes	99.9 %	61.4 MBytes	638.6 KBytes	13.0 KBytes	0	0	0	0	1.4 KBytes	0	0	0	0	0	0	0	0	0
192.168.0.1		48.9 KBytes	0.1 %	0	47.5 KBytes	0	0	0	0	0	1.4 KBytes	0	0	0	101	0	0	0	0	0
dtvro:15:c2:5e		690	0.0 %	0	0	0	0	0	0	0	690	0	0	0	0	0	0	0	0	0
cisco-linksys, llc:8a:ac:46		506	0.0 %	0	0	0	0	0	0	0	506	0	0	0	0	0	0	0	0	0
192.168.0.2		377	0.0 %	0	377	0	0	0	0	0	0	0	0	0	101	0	0	0	0	0
ff02::fb		202	0.0 %	0	202	0	0	0	0	0	0	0	0	0	202	0	0	0	0	0

Figura 46 – Dados rede interna – Ntop

As tabelas do Ntop podem apresentar os dados de várias formas e no caso da Figura 46 o Ntop está a apresentar os dados por ordem crescente de consumo de largura de banda da rede ficando assim o *host* que está a utilizar mais a rede na primeira linha. Este *host* foi responsável por 99.9% de todo o tráfego da rede o que poderia ser motivo para uma análise mais detalhada sobre essa máquina.

Analisando a linha do *host* é possível visualizar mais especificamente sobre quais protocolos é que esse tráfego foi gerado e também é possível através de uma série de símbolos obter algumas características da máquina em questão. Na primeira linha da tabela, em frente ao nome do *host*, o Ntop apresentou quatro símbolos representativos de algumas características desse *host*, o primeiro significa que a máquina tem um sistema operativo Windows, o segundo que está conectado através de uma placa de rede, o terceiro que a máquina apresenta um risco médio, e o quarto que utiliza um *software* servidor P2P (*peer to peer*), o que pode ser a razão do consumo elevado.

A segunda linha da tabela apresenta os dados sobre a máquina 192.168.0.1 que é onde está instalada o Ntop, o *software* apresentou também três símbolos sobre essa máquina. O primeiro significa que a máquina é *multi-homed*, ou seja que é usada para conectar redes, o segundo símbolo significa que a máquina tem funções de *router* e o terceiro que tem instalado um servidor DNS.

Com este exercício foi possível comprovar as funcionalidades da ferramenta, já estudadas e apresentadas, assim como integrar estas ferramentas numa solução integrada e sua aplicação (implementação) num caso de estudo (cenário), permitindo apresentar resultados práticos.

4.4 Preparação e Instalação de *Software / Hardware*

Para o projeto **TrafficMonit** foi usado um computador com processador Core 2 Duo 3 Ghz, 2GB de memória RAM, e nele foi instalado o sistema operativo ubuntu 11.10 [Ubuntu, 2011]. Após a instalação do sistema operativo foram instalados todos os pré-requisitos necessários. Foram instalados, o Nagios, Ntop, openSSH [OpenSSH, 2011] e VNC Server, estes dois últimos para que fosse possível aceder à máquina remotamente. Para que todo o tráfego da rede do DEI fosse encaminhado para a máquina do projeto foi feito um *port mirror* da porta que faz *link* ao DSI (Departamento de Sistemas Informáticos), para outra porta onde está a máquina

deste projeto que faz a monitorização do tráfego. Para que a máquina captura-se todo o tráfego incluindo o que não se destina a ela, a sua placa de rede foi colocada em modo promíscuo (ver Figura 47 –).

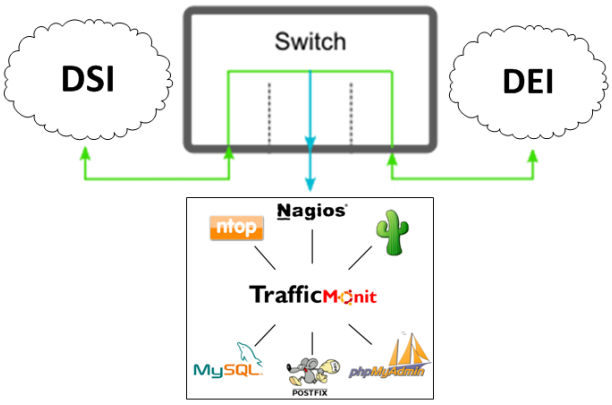


Figura 47 – Port-Mirror ao *switch* do DEI

Na realidade a máquina do projeto **TrafficMonit** possui dois interfaces de rede, a que está a fazer a monitorização do tráfego através do *port-mirror* feito no *switch* e outra para acesso à internet a qual tem o IP fixo 172.31.100.155.

4.5 Deteção de Velocidades Elevadas na Rede

Um dos objetivos do projeto **TrafficMonit** é gerar alertas quando certas situações acontecessem ou determinados limites fossem atingidos. Para tal, foi utilizado o Nagios que é um *software* já implementado na estrutura de rede do DEI-ISEP. Assim sendo, o Ntop faz uma análise da rede registando diversos valores e dependendo destes o Nagios poderá gerar alertas (por exemplo enviando um *email*). Para que isto seja possível o Nagios tem de aceder de alguma forma aos dados do Ntop. Para tal, foram seguidas duas abordagens.

O Ntop gera um conjunto de gráficos que ilustram como a rede está a ser utilizada, gera gráficos mostrando a utilização por protocolo de aplicação, por protocolo de transporte, entre outros. Para gerar estes gráficos são utilizados ficheiros RRD que ficam armazenados numa das pastas do Ntop, estando disponíveis para análise. A primeira solução para a leitura de dados do Ntop foi fazer comandos que lessem os ficheiros RRD e retornassem um valor para o Nagios. Dois dos primeiros comandos implementados foram o '*check_http*' e o '*check_emule*' e são executados periodicamente pelo Nagios (atualmente uma vez por minuto). Estes comandos verificam qual a velocidade dos dados para cada um dos protocolos. Os comandos têm 4 parâmetros de entrada, o caminho do ficheiro rrd e 3 valores numéricos para os quais o Nagios deverá reportar um '*OK*', '*Warning*' ou '*Critical*'. Uma das vantagens da utilização dos ficheiros RRD por parte do Ntop, é que estes são guardados do disco rígido mantendo assim os dados mesmo que o computador seja desligado.

Muito dos dados do Ntop não são guardados em ficheiros RRD mas apenas na memória RAM, não ficando estes valores guardados de uma forma permanente, são exemplo os IPs de origem, IPs de destino, portas utilizadas, entres muitos outros.

Para colmatar esta situação foi implementada uma solução que guarda alguns destes dados, considerados mais importantes, para uma base de dados Mysql. Para esta tarefa fez-se uso da disponibilização por parte de algumas versões do Ntop de um *socket* onde podem ser lidos os resultados das análises efetuadas. Permitindo que os dados possam ser consultados mais tarde, para fins estatísticos ou de segurança facilitando o acesso aos dados por parte do Nagios (ver Figura 48).

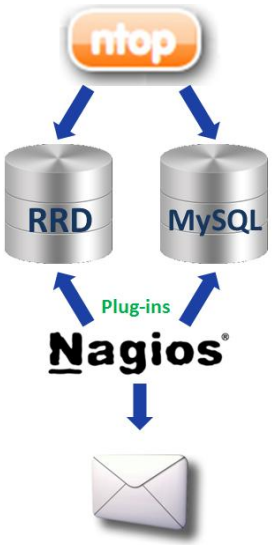


Figura 48 – Arquitetura da solução para verificação de tráfego

No entanto, durante estes desenvolvimentos foi encontrado um projeto com este *plugin* Mysql a funcionar. O projeto tem proveniência tailandesa e chama-se NtopThai [nectec, 2011].

Para detetar se alguns dos *host* está a receber dados da rede com uma velocidade excessiva, foi implementada uma função em linguagem 'C' (ver Apêndice B)que verifica a velocidade que cada utilizador está a receber dados da rede. Esta função chama-se '*check_hostIpSpeedRcvd*' e é periodicamente executada pelo Nagios que gera um alerta se necessário (ver Figura 50). A implementação desta função teve em conta algumas particularidades do funcionamento do Ntop. O Ntop apresenta os valores de *download* e *upload* para cada um dos *host*, mas estes dados referem-se ao acumulado de *bytes*, desde que foi feito um *reset* ao programa ou desde que este foi reiniciado. Ou seja, este valor não pode ser utilizado para definir se um certo utilizador está a utilizar de forma excessiva os recursos da rede, por isso a opção mais correta foi avaliar a cada minuto qual a velocidade que cada *host* a está a usar. O funcionamento da função pode ser observado na Figura 49, de seguida serão explicadas as suas tarefas:

1º - Lê da base de dados MySQL o acumulado de *bytes* que cada *host* recebeu (bytes_Rcvd);

2º - Aguarda 1 minuto e volta a ler a base de dados MySQL, o acumulado de *bytes* que cada *host* recebeu até ao momento;

3º - Fazendo a diferença das duas leituras efetuadas na base de dados, calcula o nº de *bytes* que cada um dos *hosts* recebeu no último minuto;

4º - Divide este último valor por 60, calculando assim a velocidade a que cada *host* está a utilizar a rede;

5º - Dependendo deste valor envia um retorno para o Nagios podendo este valor ser '*OK*', '*Warning*' ou '*Critical*', e dependendo dos valores que foram passados quando a função é chamada pelo Nagios;

6º - Guarda os dados calculados numa tabela Mysql (speed_Host) para posteriormente ser consultados e apresentados numa página HTML.

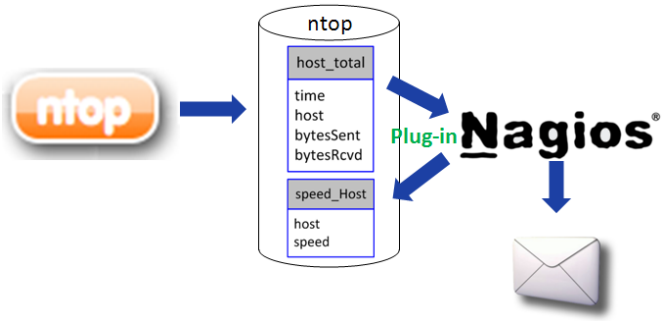


Figura 49 – Comunicação Nagios Ntop

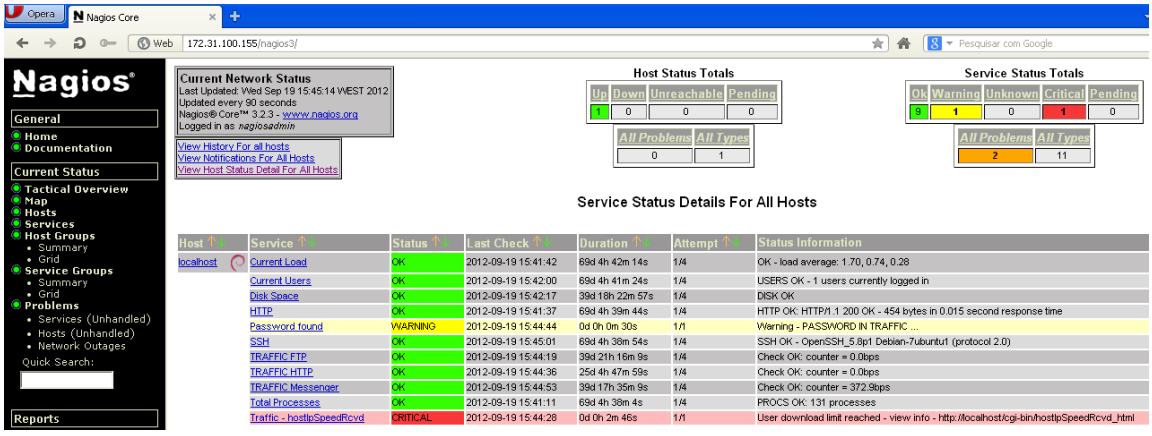


Figura 50 – Estado dos serviços do Nagios

Para que os dados das velocidades possam ser consultados foi criada uma CGI em C chamada '*hostIpSpeedRcvd_html.c*' (ver Apêndice C)que lê a tabela na base de dados e gera uma página HTML que pode ser consultada em: http://172.31.100.155/cgi-bin/hostIpSpeedRcvd_html ou na própria interface do Nagios, já que foram adicionados

novos menus para aceder às novas funcionalidades implementadas. Neste momento os alertas a serem gerados pelo Nagios são: '*Warning*' quando o tráfego for de 50kB/s a 100 kB/s e '*Critical*' quando for superior a 100kB/s (ver Figura 51).

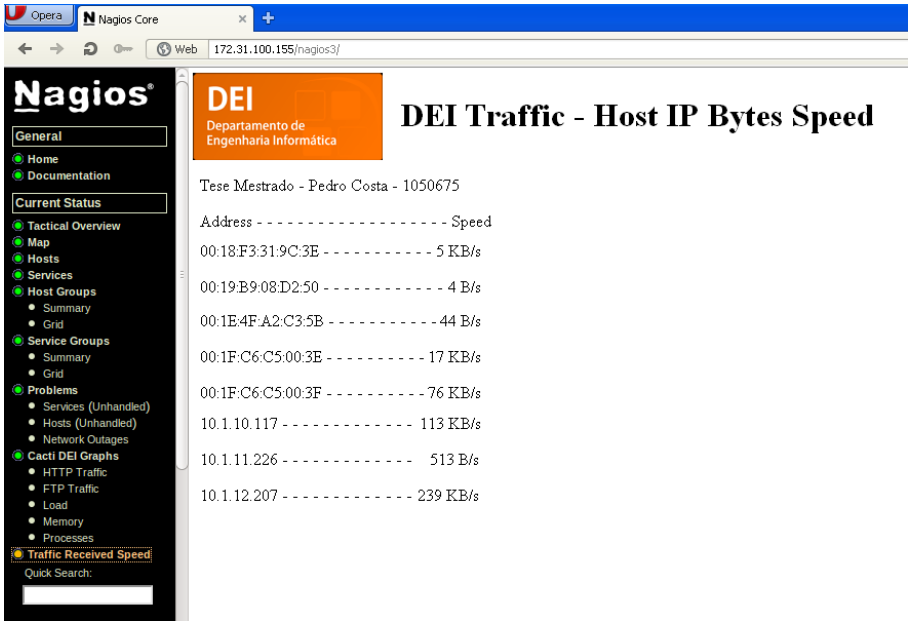


Figura 51 – Velocidade dos dados recebidos

4.6 Alertas por Eventos

O Nagios [Galstad, 2011] faz a monitorização de alguns serviços e verifica qual o estado destes: *OK*, *Warning, Critical ou Unknow*. Sempre que há mudanças entre estes estados é gerado um alerta, através de uma das formas disponíveis. Neste caso, em particular optou-se pelo serviço de *email,* já que é um serviço grátis e é o serviço atualmente utilizado pelo sistema de monitorização do DEI-ISEP. Para tal, foi instalado e configurado o servidor de *email* Postfix [Frederick P. Brooks, 2011] e foi criada uma conta de *email* para ser utilizada no envio e receção das mensagens de alerta. O *email* criado foi 'isep.dei.traffic@gmail.com' e esta informação tem que ser passada ao Nagios para que ele use esta conta na emissão dos eventos. Existe um ficheiro de configuração dos contactos do Nagios e nele para além da informação do *email*, referente ao recetor dos alertas, também é possível configurar quais os períodos semanais e em que situações estes devem ser enviados. (ver Figura 52). Após estas configurações todos os alertas são enviados para o *email* respetivo, para serem analisados pelo administrador e serem tomadas as respetivas providências (ver Figura 53).

```
define contact{
        contact_name                    root
        alias                           Root
        service_notification_period     24x7
        host_notification_period        24x7
        service_notification_options    w,u,c,r
        host_notification_options       d,r
        service_notification_commands   notify-service-by-email
        host_notification_commands      notify-host-by-email
        email                           isep.dei.traffic@gmail.com
        }
```

Figura 52 – Ficheiro contacts_nagios2.cfg do Nagios

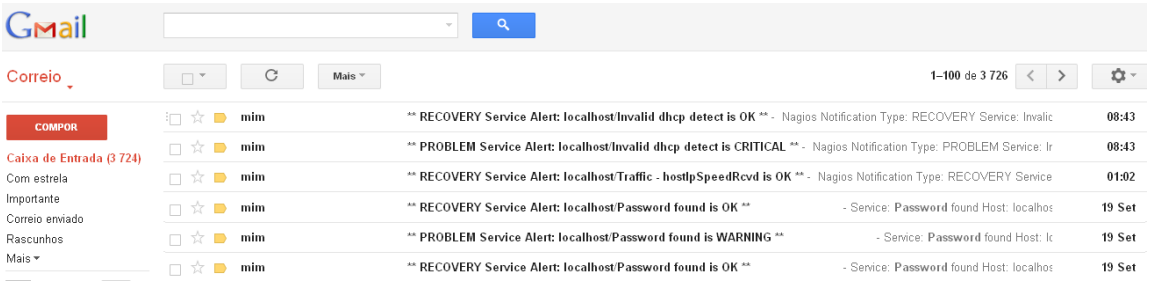


Figura 53 – Alertas recebidos por *email*

4.7 Deteção de *Passwords* na Rede do DEI

Na secção 3.7.1 foi abordado o tema confidencialidade numa rede de computadores e captura de *passwords*. Para testar estes conceitos na rede do DEI foi implementada uma solução que faz a verificação da existência de *passwords* a circular na rede. Para esse feito foi usada a ferramenta *dsniff*, que é um rastreador de tráfego especializado na captura de nomes de utilizador e *password*, sendo que é capaz de rastrear tráfego dos protocolos HTTP, SMTP, FTP, Telnet, POP, base de dados SQL entre outros protocolos. A ferramenta foi colocada a correr em *background* e a escrever as credenciais encontradas num ficheiro. Este ficheiro é constantemente verificado pelo Nagios através de um serviço chamado '*Password found*' (ver Apêndice D), este *script* foi implementado em *perl* e alerta o Nagios com um '*Warning'* no caso de encontrar alguma credencial no tráfego (ver Figura 54).

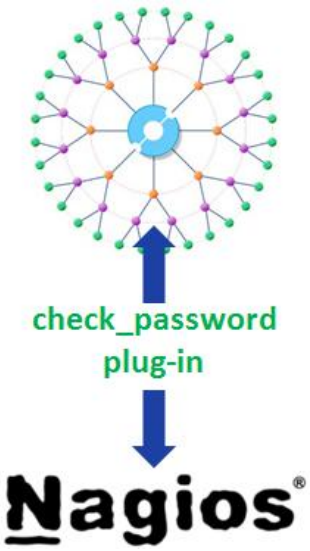


Figura 54 – Arquitetura da solução para verificação de *passwords* da rede

Como já foi referido a rede do DEI utiliza o protocolo SNMP para recolher informação dos vários dispositivos da infraestrutura. Ao solicitar essa informação aos agentes tem que ser feita uma autenticação. Muitas das credenciais detetadas pela ferramenta correspondem a credenciais usadas no processo de autenticação de cerca de vinte dipositivos monitorizados.

No entanto, o uso desta ferramenta, da forma como está implementada, pode trazer problemas de segurança, já que existe um ficheiro com as *passwords* encontradas. Este ficheiro é uma base de dados e não está no formato texto, mas a sua conversão é possível com uma opção disponibilizada pelo dsniff. Desta forma as credenciais encontradas poderiam ser lidas. Uma alteração a fazer ao *plugin* seria fazer com que sempre que uma credencial fosse detetada fosse apagado esse ficheiro, para garantir que a não existia a possibilidade deste ser consultado.

4.8 Deteção de Ataques DoS

Existe um *plugin* que pode ser instalado no Ntop que tem a capacidade de monitorizar e reportar ataques DoS indicando quais os intervenientes dos ataques. O *plugin* foi desenvolvido por S. Pukkawanna e deteta ataques do tipo: *SYN flood*, *ICMP flood*, *port scanning* e *host scanning*. Os dois primeiros ataques foram apresentados na secção 3.7.2. O *port scanning* é um conceito que consiste no teste do estado das portas de um *host* verificando se estas estão abertas, fechadas, ou em escuta. Esta técnica pode ser usada por administradores de rede para verificar políticas de segurança ou por atacantes para descobrir pontos alvo. O *host scanning* é um rastreio efetuado na rede de forma a descobrir *hosts* ativos. Tanto o *port scanning como o host scanning* podem ser realizados com a ferramenta nmap [nmap, 2012].

Após este *plugin* estar instalado foram efetuados alguns testes para verificar o seu funcionamento, um deles foi atacar a máquina de monitorização usada para esta tese com um *ping of death*. O ataque foi gerado enviando pacotes ICMP com uma dimensão excessiva de uma forma contínua. A verificação de ataques deste tipo pode ser feita pelo Ntop com uma cadência temporal configurável e quando este detetar o ataque criará o relatório (ver Figura 55).

DoS Detection

Attack Type	Attacker	Victim	Description
ICMP FLOOD	192.168.1.102	nagios-VirtualBox	Attacker attacks 2250 packet
ICMP FLOOD	nagios-VirtualBox	192.168.1.102	Attacker attacks 2250 packet

Figura 55 – Deteção do Ntop de ataque DoS - ICMP flood

4.9 Gráficos do Tráfego

Como já foi referido o Ntop guarda alguns dados sobre o tráfego na rede em bases de dados através de ficheiros RRD. Foi utilizado o *software* Cacti, para fazer a importação desses ficheiros e através destes gerar vários gráficos sobre o tráfego usado pelos vários protocolos. O Ntop requer muito processamento e por isso é essencial monitorizar variáveis, como, carga do processador, processos e memória usada. O Cacti também está a ser usado para esse fim gerando gráficos destes itens. Para facilitar a interação, foram integrados no Nagios menus para que seja possível visualizar os gráficos do tráfego. Na Figura 56 é possível visualizar os gráficos correspondentes ao tráfego HTTP da rede DEI mas vários outros protocolos também podem ser visualizados. A escolha da ferramenta Cacti para a geração dos gráficos e não de outras como por exemplo o nagiosgraph [nagiosgraph, 2011] ou nagiosgrapher [NagiosGrapher, 2012] foi porque o Cacti tem as vantagens de ser possível configurar os gráficos com imensas opções tais como: possibilidade de fazer *zoom* nos gráficos, definir quais os valores a apresentar mínimos, médios ou ambos, introduzir legendas, títulos etc.

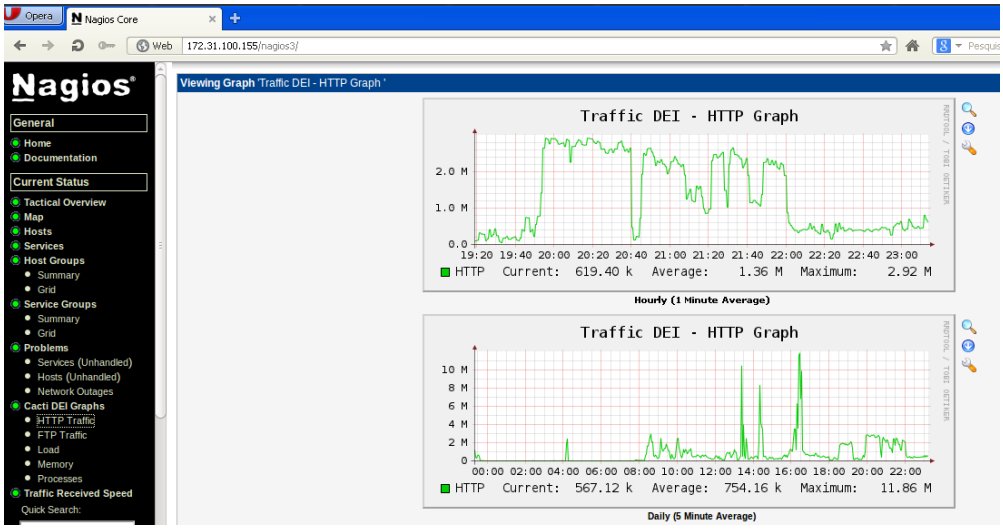
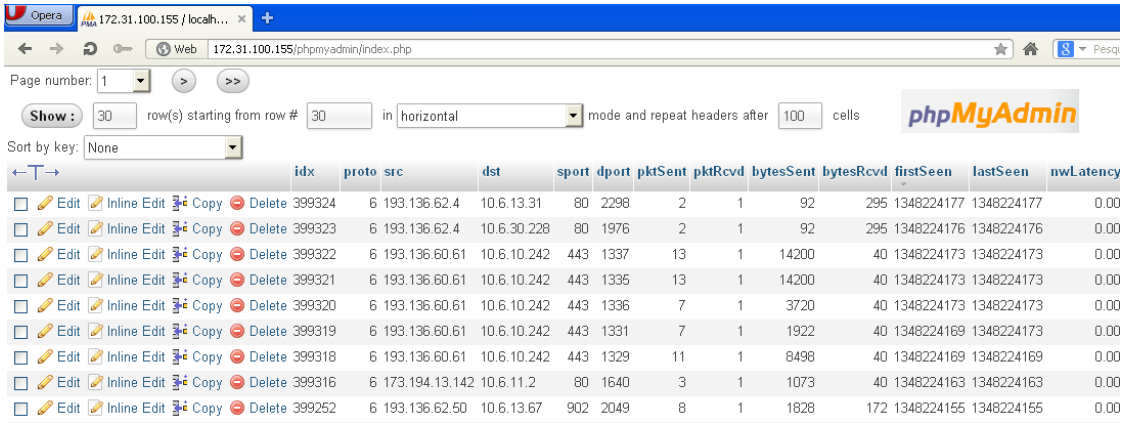


Figura 56 – Gráfico do Tráfego HTTP do DEI

4.10 Registo de Dados

As operadoras internet guardam dados sobre as sessões Internet usadas pelos seus clientes, ficando assim registado ao longo do tempo quais os IPs de origem, IPs destino, portas, entre outros dados usados na sessão. Estes dados podem ser uteis mais tarde, por exemplo, para descobrir a origem de comunicações que ponham em causa a segurança pessoal ou nacional.

No caso de o cliente ter uma rede interna constituída por vários computadores, as operadoras apenas conseguem guardar os dados das sessões até ao IP externo do cliente não ficando com qualquer registo de qual dos utilizadores da rede interna acedeu a um recurso da Internet. O projeto **TrafficMonit** implementado nesta tese monitoriza todo o tráfego Internet que passa pelo interface monitorizado, sendo possível visualizar quais os protocolos utilizados, dados sobre as sessões, contendo IPs de origem, IPs de destino, portas usadas, etc. A partir da altura que foi instalado no projeto o *plugin* Mysql estes dados ficam guardados permanentemente numa base de dados e podem ser analisados mais tarde, se necessário. Para além da verificação de quais as tendências de utilização da rede ao nível do protocolo, estes dados armazenados podem ser úteis para complementar os dados guardados pelas operadoras e ajudar traçar uma rota de uma origem até a um destino. A base de dados pode ser consultada através do *software* PhpMyadmin (ver Figura 57).



	idx	proto	src	dst	sport	dport	pktSent	pktRcvd	bytesSent	bytesRcvd	firstSeen	lastSeen	nwLatency
Edit Inline Edit Copy Delete	399324	6	193.136.62.4	10.6.13.31	80	2298	2	1	92	295	1348224177	1348224177	0.00
Edit Inline Edit Copy Delete	399323	6	193.136.62.4	10.6.30.228	80	1976	2	1	92	295	1348224176	1348224176	0.00
Edit Inline Edit Copy Delete	399322	6	193.136.60.61	10.6.10.242	443	1337	13	1	14200	40	1348224173	1348224173	0.00
Edit Inline Edit Copy Delete	399321	6	193.136.60.61	10.6.10.242	443	1335	13	1	14200	40	1348224173	1348224173	0.00
Edit Inline Edit Copy Delete	399320	6	193.136.60.61	10.6.10.242	443	1336	7	1	3720	40	1348224173	1348224173	0.00
Edit Inline Edit Copy Delete	399319	6	193.136.60.61	10.6.10.242	443	1331	7	1	1922	40	1348224169	1348224173	0.00
Edit Inline Edit Copy Delete	399318	6	193.136.60.61	10.6.10.242	443	1329	11	1	8498	40	1348224169	1348224169	0.00
Edit Inline Edit Copy Delete	399316	6	173.194.13.142	10.6.11.2	80	1640	3	1	1073	40	1348224163	1348224163	0.00
Edit Inline Edit Copy Delete	399252	6	193.136.62.50	10.6.13.67	902	2049	8	1	1828	172	1348224155	1348224155	0.00

Figura 57 – Dados sobre as sessões internet

4.11 Deteção de Servidores DHCP Indevidos

Na grande maioria das vezes que um computador se liga a uma rede obtém o seu IP de uma forma dinâmica, tal acontece porque existe na infraestrutura um servidor DHCP (ver Figura 58). Em certas situações acontece que um segundo servidor DHCP é ligado de uma forma não autorizada e a partir desse momento as novas máquinas ligadas na rede poderão ser servidas por este serviço. Isto pode acontecer acidentalmente ou propositadamente, o primeiro caso poderá acontecer porque um cliente da rede ao se ligar à infraestrutura esqueceu-se de desativar no seu computador um servidor DHCP que usava em outra situação, ou então alguém ligou um outro *router* com o servidor DHCP ativo. Nestas situações os clientes que tiveram obtido o IP destes servidores não terão um encaminhamento correto dos seus pacotes e não terão acesso aos recursos pretendidos. Quando os servidores DHCP são ligados à rede por parte dos clientes de uma forma propositada, pode existir intenção de efetuar um ataque. Nesta situação o servidor vai responder a todas as solicitações DHCP colocando o seu próprio IP como sendo o *gateway* padrão. Dado isto, todas as solicitações de acesso à rede efetuadas por parte dos clientes serão encaminhadas pelo atacante para o *gateway* real e as respostas serão devolvidas aos clientes, mas todo o tráfego da rede gerado para o cliente ficará disponível para análise pelo atacante (ver Figura 59).

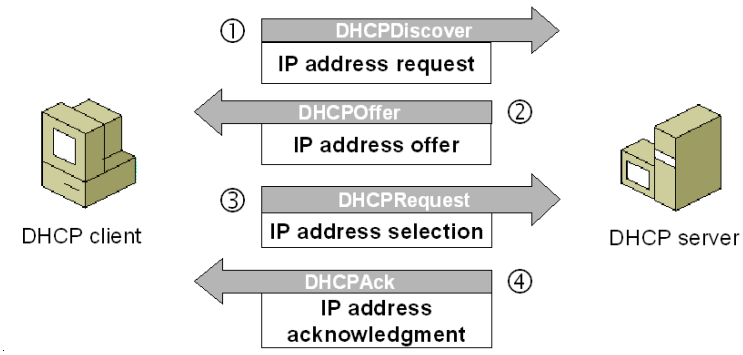


Figura 58 – Funcionamento do protocolo DHCP (baseado em [Microsoft, 2011])

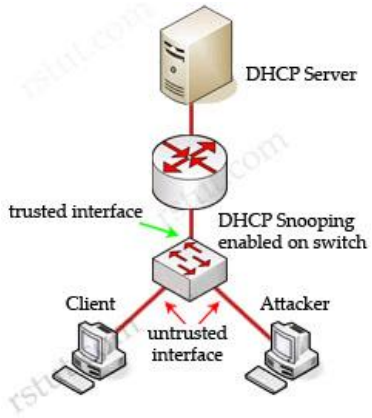


Figura 59 – Ataque com servidor DHCP indevido (baseado em [Security, 2011])

Para detetar estas situações foi implementando um *plugin* para o Nagios chamado '*Invalid dhcp detect*' (ver Apêndice E) que deteta servidores DHCP indevidos. Ele tem como base um programa desenvolvido por Andras Kovacs e Tamas Sule sobre licença GPL e foi modificado para ter os requisitos necessários. A Rede do DEI tem disponíveis 3 servidores DHCP os quais tem os endereços IP: 10.3.0.1, 10.6.0.1, 172.31.0.1 e os endereços MAC: 00:19:b9:f9:c4:32, 00:07:e9:31:81:c3, 00:18:f3:64:53:36, respetivamente. A função do *plugin* é verificar todos os servidores DHCP que estão na rede e se existir outro que não os mencionados anteriormente será gerado um alerta (ver Figura 60).

Current Status:	OK (for 0d 0h 57m 38s)
Status Information:	Dhcp server detected: mac=00:19:b9:f9:c4:32, ip=10.3.0.1 - DHCP Server Recognized Dhcp server detected: mac=00:07:e9:31:81:c3, ip=10.6.0.1 - DHCP Server Recognized Dhcp server detected: mac=00:18:f3:64:53:36, ip=172.31.0.1 - DHCP Server Recognized
Performance Data:	
Current Attempt:	1/1 (HARD state)
Last Check Time:	2012-09-25 02:42:53
Check Type:	ACTIVE
Check Latency / Duration:	0.088 / 2.081 seconds
Next Scheduled Check:	2012-09-25 02:43:53
Last State Change:	2012-09-25 01:45:53
Last Notification:	N/A (notification 0)
Is This Service Flapping?	N/A
In Scheduled Downtime?	NO
Last Update:	2012-09-25 02:43:28 (0d 0h 0m 3s ago)

Figura 60 – Informação sobre o estado do serviço '*Invalid dhcp detect*'

Uma forma de bloquear os servidores indevidos na rede é tirar proveito de uma opção que alguns *switchs* possuem chamada DHCP *snooping* que é uma série de técnicas que garantem a segurança de uma infraestrutura DHCP garantindo a integridade IP na camada 2. Este atua como uma *firewall* entre *hosts* não confiáveis e servidores DHCP e também providencia uma forma de diferenciar entre as interfaces não confiáveis ligadas ao utilizador final e as interfaces confiáveis ligadas ao servidor DHCP ou a outro *switch*. Assim sendo, as portas podem ser configuradas como *trusted* (confiável) ou *untrusted* (não confiável) só podendo as primeiras responder às solicitações de requisição. As portas configuradas como *untrusted* (não confiável) serão desabilitadas caso recebam respostas para solicitações de requisição DHCP [DHCP, 2011] [Snooping, 2011] [IOS, 2010].

O DHCP *snooping* trabalha com informações de um servidor DHCP para: rastrear a localização física dos *hosts*, garantir que os *hosts* apenas usam os IPs atribuídos aos próprios, e garantir que apenas servidores DHCP autorizados são acessíveis. Com este recurso, apenas os endereços IP que estão numa lista branca podem aceder à rede, sendo que esta lista é configurada ao nível das portas do *switch* e o servidor DHCP é quem faz a gestão do controlo de acessos. Apenas endereços IP específicos e com endereços específicos MAC, em portas específicas, podem aceder à rede IP [DHCP, 2011] [Snooping, 2011] [IOS, 2010].

O DHCP *snooping* pode prevenir que atacantes adicionem os seus próprios servidores DHCP na rede com seus próprios servidores, o que poderia causar o mau funcionamento da rede ou mesmo o seu controlo. Também é um componente importante na defesa do ataque ARP *spoofing*. A segurança ARP verifica o endereço IP no campo *Source Protocol Address* dos pacotes ARP. Se esse endereço IP não for um endereço ao qual o DHCP *snooping* gravou como estando em uso por uma máquina ligada à porta ARP, o pacote ARP será descartado [DHCP, 2011] [Snooping, 2011] [IOS, 2010].

5 Conclusões e Trabalho Futuro

A massificação da utilização das tecnologias de informação e da Internet para os mais variados fins, e nas mais diversas áreas, levantou problemas de gestão das infraestruturas de informática, ímpares até ao momento. Paralelamente, o aumento do grau de complexidade das redes e do seu tamanho exige o emprego de um sistema de gestão que proporcione qualidade de serviço, pro-atividade, diferenciação de tráfego e o suporte multifacetado de serviços, assim como integração com o processo de serviços e negócio.

Pretendia-se com este trabalho efetuar um estudo de vários protocolos, ferramentas de gestão e de classificação de tráfego. Paralelamente, pretendia-se instalar e configurar ferramentas de gestão e classificação de tráfego num cenário real que permitisse apresentar resultados práticos. O cenário escolhido foi a rede atual do Departamento de Engenharia do Instituto Superior de Engenharia do Porto.

O projeto **TrafficMonit** permite monitorizar o tráfego da rede e caso existam situações que coloquem em causa a segurança ou desempenho da mesma gera um conjunto de alertas que permitem que sejam tomadas as medidas necessárias para ultrapassar os problemas detetados, permitindo que os utilizadores da rede não sejam prejudicados, quer em termos dos serviços disponíveis, quer na performance da rede.

O Projeto agrupa um conjunto de funcionalidades que permitem efetuar análise e classificação de tráfego. Agrega, ainda, a informação obtida para que esta seja verificada pelo Nagios. O Nagios irá gerar, se necessário, eventos. Foram implementadas funcionalidades que visam melhorar a segurança da rede tais como: deteção de ataques DoS, deteção de servidores DHCP indevidos, e deteção de *passwords* na rede. Também foram implementadas outras funcionalidades que tem como objetivo melhorar a qualidade de serviço da rede tais como: deteção de velocidades elevadas na rede, gerar gráficos do tráfego, separado por protocolo e registo dos dados relativos às ligações.

Algumas das conclusões obtidas são:

- O projeto TrafficMonit, constituído por um conjunto de ferramentas *Open Source* revela-se ser muito flexível;
- No entanto, a implementação com várias ferramentas revelou-se complexa relativamente à sua configuração inicial;
- A integração das várias interfaces (ferramentas) num ambiente único facilita a sua utilização;
- A disponibilidade de informação histórica relativa ao tráfego num formato gráfica facilita a análise dos dados;

- A deteção de eventos capazes de colocar em causa a segurança da rede, permite gerar alertas e a possibilidade de que sejam aplicadas medidas de correção;
- A deteção de tráfegos anormais pode ser utilizada para avaliar quais as melhores técnicas a usar para evitar problemas de congestionamento;
- O registo de dados relativos às informações sobre as ligações pode ser usado para identificar a origem de comunicações, que ponham em causa a segurança pessoal ou da rede.

5.1 Trabalho Futuro

Neste momento a função de monitorização de tráfego não está a ser feita em toda a rede do departamento de informática do ISEP, apenas num dos pisos do edifício. Um trabalho futuro seria precisamente fazer estender esta funcionalidade a toda a rede. Permitindo, que o tráfego de todos os pisos (que constituem o Departamento de Engenharia Informática - DEI) fosse analisado. Como referido o DEI já tinha disponível a função de monitorização implementada através da ferramenta Nagios. O sistema proposto neste trabalho também implementa a função de monitorização através do Nagios, ou seja existem duas instâncias desta ferramenta na rede do DEI, para fazer monitorização. Convém no entanto, referir que esta decisão foi tomada para que não se perdesse a garantia de que a monitorização era feita enquanto este trabalho evoluía. Pelo que, uma melhoria para o sistema seria exportar as funções do Nagios da máquina deste projeto para apenas termos uma ferramenta.

A proposta anterior, pelo facto de a máquina que faz a classificação do tráfego e a que gera os alertas passarem a estar em máquinas distintas implica que se disponibilize um mecanismo capaz de executar os *plugins* remotamente.

O tráfego está a ser analisado, classificado e registado, seria interessante futuramente analisar as tendências da rede registadas através da verificação dos dados o que permitira avaliar que futuros melhoramentos se deveriam considerar. Por exemplo, caso fossem detetadas situações de congestionamento da rede, poderia ser ponderada a necessidade da aplicação de algumas técnicas de QoS, com vista a melhorar o funcionamento da rede.

Para testar o sistema proposto foram desenvolvidos alguns *plugins* que permitiam testar diversas situações. Estes foram desenvolvidos apenas para demonstrar o funcionamento do sistema em si e quando estes foram testados não foram feitos grandes ajustes ao código, mas com certeza estes poderia ter várias otimizações de forma a obter melhores resultados.

Referências

[André,2011] [Online] Disponível em: http://andredeo.blogspot.pt/2011/09/ [Acedido em 07 12 2011].

[Analyser, 2011] NetFlow Analyser - NetFlow Analysis - NetFlow Collector – NetFlow Monitor. [Online] Disponível em: http://www.netflow.co.uk/ [Acedido em 03 12 2011].

[Andrey Vedana Andreoli, 2006] L. M. B. L. M. R. T., PRÁTICA EM SEGURANÇA DE REDES

[Antoniou, S., 2009] The PING of Death and Other DoS Network Attacks [Online] Disponível em: http://www.trainsignal.com/blog/ping-of-death-and-dos-attacks [Acedido em 23 09 2012].

[Cacti, 2009] Cacti - Open Source Performance Monitoring.

[Case, 1995] Managing Local Area Networks. San Francisco: McGraw-Hill.

[Castro, 2008] Monitorização e Detecção Automática de Anomalias.

[Cisco, 2005] INTRODUCTION TO SNMP AND MIBs - SESSION NMS-1101. : Cisco Systems.

[Cisco, 2010] Cisco IOS Software Configuration Guide—Release 12.1(12c)EW.

[Cisco_IOS_11.3, 2012] Cisco_IOS_11.3, 2012. SNMP Inform Requests. [Online] Disponível em: http://www.cisco.com/en/US/docs/ios/11_3/ /feature/guide/snmpinfm.html [Acedido em 27 09 2012].

[ComputerWorld, 2010] ComputerWorld. ComputerWorld.

[debian.org, 2012] *debian.org.* [Online] Disponível em: http://www.debian.org/ [Acedido em 03 01 2012].

[DEI, 2012] *Monitorização da Rede e Serviços.* [Online] Disponível em: http://nagios.dei.isep.ipp.pt/ [Acedido em 12 09 2012].

[DHCP, 2011] DHCP snooping. [Online] Disponível em: http://en.wikipedia.org/wiki/DHCP_snooping [Acedido em 10 09 2012].

[Dinangkur Kundu, 2009] Cacti 0.8 Network Monitoring. Birmingham, B27 6PA, UK.: Packt Publishing Ltd.

[ehrizo,2011] [Online] Disponível em: http://ehrizo.wordpress.com/?s=RMON [Acedido em 15 12 2011].

[Flannagan, 2001] Administering Cisco QoS for IP Networks. United States of America: Syngress Publishing, Inc.

[Frederick P. Brooks, 2011] *postfix.org.* [Online] Disponível em: http://www.postfix.org/ [Acedido em 03 11 2011].

[Galstad, 2011] *Nagios.* [Online] Disponível em: http://www.nagios.org/ [Acedido em 16 11 2011].

[Hegering, 1999] Integrated Management of Networked Systems. San Francisco, California: Morgan Kaufmann Publishers.

[Kurose, 2010] Computer networking fifth edition a top-down approach. Editora Pearson.

[Leobino Sampaio, 2002] RNP - Projeto piloto de medições.

[Max Schubert, 2008] Nagios 3 Enterprise Network Monitoring. United States of America: Syngress.

[Maxwell, 1999] *UNIX network management tools.* United States of America: McGraw-Hill.

[Microsoft, 2011] *Chapter 6 - Dynamic Host Configuration Protocol.* [Online] Disponível em: http://technet.microsoft.com/en-us/library/bb727003.aspx [Acedido em 10 10 2012].

[mysql, 2011] *mysql.* [Online] Disponível em: http://www.mysql.com/ [Acedido em 10 12 2011].

[Monteiro & Boavida, 2000] *Engenharia de Redes Informáticas.* s.l.:FCA - Editora de Informática.

[Muller, 1996] Network Planning, Procurement, & Management. s.l.:McGraw-Hill.

[nagiosgraph, 2011] *nagiosgraph - data collection and graphing for nagios.* [Online] Disponível em: http://nagiosgraph.sourceforge.net/ [Acedido em 26 06 2012].

[NagiosGrapher, 2012] *NagiosGrapher.* [Online] Disponível em: http://sourceforge.net/projects/nagiosgrapher/ [Acedido em 28 06 2012].

[nagiosQL, 2012] *NagiosQL.* [Online] Disponível em: http://www.nagiosql.org/ [Acedido em 13 04 2012].

[nagvis, 2011] *[NagVis.* [Online] Disponível em: http://www.nagvis.org/ [Acedido em 03 11 2011].

[nectec, 2011] *nectec.* [Online] Disponível em: http://wiki.nectec.or.th/ntl/Project/NtopPublic [Acedido em 12 01 2012].

[Netflow, 2012] *Netflow.* [Online] Disponível em: http://commons.wikimedia.org/w/index.php?title=File:Netflow_architecture_en.svg&page=1

[net-snmp, 2012] *net-snmp.* [Online] Disponível em: http://www.net-snmp.org/ [Acedido em 10 08 2012].

[nmap, 2012] *nmap, 2012. nmap.* [Online Disponível em: http://nmap.org/ [Acedido em 23 09 2012].

[Nogueira, 2004] Gestão de Redes - Um Serviço de Valor Acrescentado, Porto.

[Ntop, 2011] *ntop.* [Online] Disponível em: http://www.ntop.org/ [Acedido em 26 11 2011].

[ntop, 2012] *ntop - Traffic analysis with NetFlow™ and sFlow™ support.* [Online] Disponível em: https://www.ntop.org/products/ntop/ [Acedido em 09 03 2012].

[NRPE, 2012] *NRPE.* [Online] Disponível em: http://nagios.sourceforge.net/docs/3_0/addons.html [Acedido em 29 09 2012].

[Oetiker's, 2011] *MRTG - Tobi Oetiker's MRTG - The Multi Router Traffic Grapher.* [Online] Disponível em: http://oss.oetiker.ch/mrtg/ [Acedido em 06 07 2012].

[Oliveira, 2005] Gestão Centralizada de Parques Informáticos. Porto.

[OpenSSH, 2011*]* [Online] Disponível em: www.openssh.com [Acedido em 08 10 2012].

[Pan, H., 1998] *SNMP - Based ATM Network.* s.l.:Artech House Publishers.

[phpmyadmin, 2008] *phpmyadmin.* [Online] Disponível em: http://www.phpmyadmin.net/home_page/index.php [Acedido em 05 04 2012].

[Plugins, 2011] *Nagios Plugins.org.* [Online] Disponível em: http://nagiosplugins.org/ [Acedido em 07 11 2011].

[postfix, 2008] *postfix.* [Online] Disponível em: http://www.postfix.org/ [Acedido em 12 12 2012].

[postgresql, 2011] *postgresql.org.* [Online] Disponível em: http://www.postgresql.org/ [Acedido em 20 12 2011].

[RMON, 2011] *RMON - SNMP Remote Monitoring.* [Online] Disponível em: https://www.classle.net/content-page/rmon-snmp-remote-monitoring [Acedido em 05 03 2012].

[RMON2, 2011] *RMON2 Basics.* [Online] Disponível em: http://www.pulsewan.com/data101/rmon2_basics.htm [Acedido em 05 04 2012].

[RMON, 2003/2004] CONFIGURACIÓN DE SNMP Y RMON EN ROUTERS Y SWITCHES CISCO.

[rrdtool, 2011] *rrdtool.* [Online] Disponível em: http://oss.oetiker.ch/rrdtool/ [Acedido em 06 08 2012].

[Ruzicka, 2003] Network Management with Nagios, Netsaint's Successor. *linux-magazine.*

[Security, 2011] *Security Questions.* [Online] Disponível em: http://www.rstut.com/ccie-written/security-questions

[Acedido em 10 10 2012].
[sFlow.org, 2003] Traffic Monitoring using sFlow.
[Snooping, 2011] *DHCP Snooping.* [Online]
Disponível em: http://moises-araujo.blogspot.pt/2010_09_01_archive.html
[Acedido em 10 09 2012].
[SonicWALL, 2009] Denial of Service Attacks:
An Emerging Vulnerability for the "Connected" Network,
[Systems, 2012] Systems, C., 2012. SNMPv3 - Cisco Systems. [Online] Disponível em:
http://www.cisco.com/en/US/docs/ios/12_0t/12_0t3/feature/guide/Snmp3.html
[Acedido em 23 09 2012].
[tcpipguide,2011] Disponível em: http://www.tcpipguide.com/free/
t_TCPIPRemoteNetworkMonitoringRMON-2.htm [Acedido em 15 12 2012].
[Ubuntu, 2011] *Ubuntu.* [Online] Disponível em: http://www.ubuntu.com/
[Acedido em 12 10 2011].
[unix, 2012] *unix.* [Online] Disponível em: http://www.unix.org/ [Acedido em 28 12 2011].
[Urban, 2011] CACTI - Learn Cacti and design a robust Network Operations Center.
Birmingham, B27 6PA, UK.: Packt Publishing.
[Waldbusser, 1991] Remote network monitoring management information base.
Patente N.º IETF RFC 1271 S.
[Wiki, 2012] *Gerência de redes.* [Online] Disponível em: http://pt.wikipedia.org/wiki/
Ger%C3%AAncia_de_redes#
Comparativo_dos_Modelos_de_Ger.C3.AAncia_de_Redes
[Acedido em 02 02 2012].
[wireshark, 2011] *[wireshark.* [Online]
Disponível em: http://www.wireshark.org/
[Acedido em 12 12 2012].
[Wireshark, 2011] *Capture Setup Ethernet - The Wireshark Wiki.* [Online]
Disponível em: http://wiki.wireshark.org/CaptureSetup/Ethernet
[Acedido em 12 03 2012].
[Zúquete, 2006] Segurança em Redes Informáticas. Lisboa:
FCA – Editora de Informática, Lda.

Apêndice A:

A tabela seguinte compara os avançados recursos do Sflow com soluções baseadas em RMON e NetFlow da Cisco. RMON (4 *groups*) refere-se às quatro funções básicas do RMON (Estatísticas, Histórico, Alarmes e Eventos) que são muitas vezes incorporadas nos interfaces dos *switchs*. RMON II refere-se a implementações completas de RMON II, estes são tipicamente fornecidos sob a forma de uma placa adicional ou uma sonda de *hardware*.

	RMON (4 groups)	RMON II	NetFlow®	sFlow®
Packet Capture	N	Y	N	P
Interface Counters	P	P	N	Y
Protocols				
Packet headers	N	P	N	Y
Ethernet/802.3	N	Y	N	Y
IP/ICMP/UDP/TCP	N	Y	Y	Y
IPX	N	Y	N	Y
Appletalk	N	Y	N	Y
Layer 2				
Input/output interface	N	N	Y	Y
Input/output priority	N	N	N	Y
Input/output VLAN	N	N	N	Y
Layer 3				
Source subnet/prefix	N	N	Y	Y
Destination subnet/prefix	N	N	Y	Y
Next hop	N	N	Y	Y
BGP 4				
Source AS	N	N	P	Y
Source Peer AS	N	N	P	Y
Destination AS	N	N	P	Y
Destination Peer AS	N	N	P	Y
Communities	N	N	N	Y
AS Path	N	N	N	Y
Real-time data collection	Y	Y	P	Y
Configuration				
Configurable without SNMP	N	N	Y	Y
Configurable via SNMP	Y	Y	N	Y
Low Cost	Y	N	N	Y
Scalable (switch interfaces/collector)	P	N	N	Y
Wire-speed	Y	P	P	Y

N	Feature not supported
P	Feature partially supported. Either the feature is incomplete or can only be enabled by disabling other features.
Y	Fully supported

Apêndice B:

```
/*
 *         Tese de Mestrado - Pedro Costa - 1050675 - 2011-2012
 *
 * Ferramentas de Gestão de Redes e Classificação de Tráfego - Estudo de um Caso
 *
 *         ISEP - Instituto Superior de Engenharia do Porto
 *
 *                   Plugin - check_hostipSpeedRcvd
 */

#include <my_global.h>
#include <mysql.h>
#include <stdio.h>
#include <string.h>
#include <sys/stat.h>
#include <stdlib.h>


void extract(char *,char *,char *,int,int);

int main(int argc, char **argv)
{

MYSQL *conn;
MYSQL_RES *result;
MYSQL_ROW row;
int num_fields;
int i,j=0,k=0;
char *dados1[10000][3];
char *dados2[10000][3];
char *dados_geral[10000][3];
int tem=0;
char query[200];
int speed_ok_limit;
int speed_critical_limit;

if(argc != 3 ) // Testa se o número de parametros fornecidos está correcto


{
    printf("Erro! - Parâmetros Incorrectos\n");
       exit(0);
}


else
{
speed_ok_limit = atoi(argv[1]);//lê da linha de comandos o valor de ok
speed_critical_limit = atoi(argv[2]);//lê da linha de comandos o valor de Critical

}
```

```
  conn = mysql_init(NULL);
  mysql_real_connect(conn, "localhost", "root", "nagios", "ntop", 0, NULL, 0);

  mysql_query(conn, "SELECT `time` , `host` , `ipBytesRcvd` FROM `Table_host_total`
WHERE  `time` > DATE_SUB(NOW(), INTERVAL 1 MINUTE) ");
  result = mysql_store_result(conn);
  num_fields = mysql_num_fields(result);

  while ((row = mysql_fetch_row(result)))
  {

        for(i = 0; i < num_fields; i++)
        {
               dados1[j][i]=row[i];//coloca os dados num array

        }
         j=j+1;
    }

int n_dados_atuais=k;
// dados com 1 minuto
j=0;

   mysql_query(conn, "SELECT `time` , `host` , `ipBytesRcvd` FROM `Table_host_total`
WHERE  `time` > DATE_SUB(NOW(), INTERVAL 2 MINUTE) and  `time` < DATE_SUB(NOW(),
INTERVAL 1 MINUTE) ");
  result = mysql_store_result(conn);
  num_fields = mysql_num_fields(result);


  while ((row = mysql_fetch_row(result)))
  {

        for(i = 0; i < num_fields; i++)
        {
         dados2[j][i]=row[i];//coloca os dados num array

        }

     j=j+1;

  }

int n_dados_5_min=k;
int b;

       for (i=0 ;i< n_dados_atuais; i++ ) //colocar os dados1 em dados geral
              {
              dados_geral[i][0]=dados1[i][1];
              dados_geral[i][1]=dados1[i][2];
              dados_geral[i][2]="0";

       }
```

```
int f=n_dados_atuais;

                    //colocar no array dados_geral
       for (i=0 ;i<f; i++ )
              {
                    for (j=0 ;j< n_dados_5_min; j++ )
                           {
                                  b=strcmp(dados_geral[i][0],dados2[j][1]);
                                  if (b==0)     {
                                         dados_geral[i][2]=dados2[j][2];
                                                }
                           }
              }

// colocar no array dados_geral, os dados do array dados2 que não estavam em dados1

int m=n_dados_atuais;
for (i=0 ;i< n_dados_5_min; i++ )
              {
                    tem=0;
                    for (j=0 ;j< m; j++ )
                           {

                                  b=strcmp(dados2[i][1],dados_geral[j][0]);
                                  if (b==0)     {
                                  tem=1;

                                                }
                           }


              if (tem==0) {//se ainda não tiver os dados em dados_geral, acrescenta
ao vector
              dados_geral[n_dados_atuais][2]=dados2[i][2];
              dados_geral[n_dados_atuais][0]=dados2[i][1];
              dados_geral[n_dados_atuais][1]="0";
              n_dados_atuais=n_dados_atuais+1;
                    }
              }

int totalbytes[10000];
int velocidade[10000];
int nagios_state=5;
int velocidade_max=0;

              for (i=0 ;i< n_dados_atuais; i++ ) {//calcula os bytes recebidos
durante 1 minuto
              totalbytes[i]=atoi(dados_geral[i][1])-atoi(dados_geral[i][2]);
              if (totalbytes[i] < 0) totalbytes[i]=0; //se der negativo coloca igual
a 0
              velocidade[i]=totalbytes[i]/(60);
              if(velocidade_max < velocidade[i]) velocidade_max=velocidade[i];
              }
              if (velocidade_max > speed_critical_limit)  {
```

```
                    nagios_state=2;//retorno para nagios Critical
                    printf("User download limit reached - view info -
http://localhost/cgi-bin/hostIpSpeedRcvd_html");//Esta mensagem aparece no
descritivo do nagios
                    printf("\n");
                                        }
             if (velocidade_max < speed_ok_limit) {
                    nagios_state=0;//retorno para nagios OK
                    printf("Check Ok - User download limit Ok - view info -
http://localhost/cgi-bin/hostIpSpeedRcvd_html");//Esta mensagem aparece no
descritivo do nagios
                    printf("\n");
                                        }
             if ((velocidade_max > speed_ok_limit) && (velocidade_max <
speed_critical_limit)){
                    nagios_state=1;//retorno para nagios Warning
                    printf("User download limit Warning - view info -
http://localhost/cgi-bin/hostIpSpeedRcvd_html");//Esta mensagem aparece no
descritivo do nagios
                    printf("\n");
                                        }
             printf("Tese Mestrado - Pedro Costa - 1050675");
             printf("\n");
             printf("DEI Traffic - Host IP Speed");
             printf("\n");
             printf("\n");
             printf("Address                    Speed");
             printf("\n");

             //Limpa a Tabela Table_Speed_Host antes de colocar lá os dados
             sprintf(query, "DELETE FROM Table_Speed_Host; ");
             mysql_query(conn, query);
             result = mysql_store_result(conn);

             for (i=0 ;i< n_dados_atuais; i++ ) {//coloca os dados IP  e Speed na
base de dados
             sprintf(query, "INSERT INTO Table_Speed_Host(host, speed) values('%s',
%d);",dados_geral[i][0],velocidade[i]);
             //printf("%s \n",query);     //ver as query que vão para a BD
             mysql_query(conn, query);
             result = mysql_store_result(conn);
                                        }
             for (i=0 ;i< n_dados_atuais; i++ ) {//Imprime os dados das velocidades
para cada host

             if (velocidade[i] > 1000) {
             printf("%s \t- %d kB/s\n",dados_geral[i][0],velocidade[i]/1000);
                                  }
             else{
             printf("%s \t- %d B/s\n",dados_geral[i][0],velocidade[i]);
             }
             }

 mysql_free_result(result);
 mysql_close(conn);
```

```
                if (nagios_state==2) return (2);
                if (nagios_state==0) return (0);// return para o nagios OK
                if (nagios_state==1) return (1);

}
```

Apêndice C:

```
/*
 *         Tese de Mestrado - Pedro Costa - 1050675 - 2011-2012
 *
 * Ferramentas de Gestão de Redes e Classificação de Tráfego - Estudo de um Caso
 *
 *         ISEP - Instituto Superior de Engenharia do Porto
 *
 *                   CGI - Gera página para visualização dos Hosts
 */

#include <my_global.h>
#include <mysql.h>
#include <stdio.h>
#include <string.h>
#include <sys/stat.h>
#include <stdlib.h>

int main(int argc, char **argv){

MYSQL *conn;
MYSQL_RES *result;
MYSQL_ROW row;
int num_fields;
int i=0,j=0,k=0;
har *data_speed[10000][1];
char *dados_geral[10000][3];
char buffer[100]="";


conn = mysql_init(NULL);
  mysql_real_connect(conn, "localhost", "root", "nagios", "ntop", 0, NULL, 0);

  mysql_query(conn, "SELECT * FROM Table_Speed_Host; ");
  result = mysql_store_result(conn);
  num_fields = mysql_num_fields(result);

printf("Content-type:\n\n");
printf("<HTML>");
printf("<HEAD>");
printf("<TITLE>DEI Traffic</TITLE>");

printf("</HEAD>");
printf("<BODY>");
printf("<H1>DEI Traffic - Host IP Bytes Speed</H1>");
printf("Tese Mestrado - Pedro Costa - 1050675<P>");
printf("Address - - - - - - - - - - - - - - - - - - - - Speed<P>");



int m;
  while ((row = mysql_fetch_row(result)))
```

```
  {

        for(i = 0; i <num_fields; i++)
        {
        m=i+1;
         data_speed[j][0]=row[i];//coloca os dados num array
        data_speed[j][1]=row[m];//coloca os dados num array
        if (atoi(data_speed[j][1])>1000) {
        sprintf(buffer, "%s    - - - - - - - - - - -      %d
\n",row[i],atoi(data_speed[j][1])/1000);
        printf(buffer);
        printf(" KB/s");
        }
        else
        {
        sprintf(buffer, "%s    - - - - - - - - - - -      %s   \n",row[i],row[m]);

        printf(buffer);
        printf(" B/s");

        }
        printf("<P>");
        i=i+1;
        }


    j=j+1;

  }

printf("</BODY>");
printf("</HTML>");
mysql_free_result(result);
mysql_close(conn);
return 0;
}
```

Apêndice D:

```
/*
 *         Tese de Mestrado - Pedro Costa - 1050675 - 2011-2012
 *
 * Ferramentas de Gestão de Redes e Classificação de Tráfego - Estudo de um Caso
 *
 *         ISEP - Instituto Superior de Engenharia do Porto
 *
 *                   Plugin - check_pass
 */

#!/usr/bin/perl -w
use File::Compare;
use File::Copy;
use strict;

my $OKAY     = 0;
my $WARNING  = 1;
my $CRITICAL = 2;
my $UNKNOW   = 3;

my $test =compare("/home/nagios/pass_net_rt.log","/home/nagios/pass_net_bk.log");#
compara o ficheiro de recolha de pass com o ficheiro guardado 1 minuto antes

if ($test == 0){ #se forem iguais ...
       copy("/home/nagios/pass_net_rt.log","/home/nagios/pass_net_bk.log") or die
"Copy failed: $!";    #copia o ficheiro de recolha de pass para outra cópia
       printf("Check OK - No Password in Traffic ...");
       printf("\n");
       exit($OKAY);
       }

if ($test != 0){
       copy("/home/nagios/pass_net_rt.log","/home/nagios/pass_net_bk.log") or die
"Copy failed: $!";
       printf("Warning - PASSWORD IN TRAFFIC ...");
       printf("\n");
       exit($WARNING);
       }
```

Apêndice E:

```
/* /*
 * Dhcpdetector
 *
 * Copyright 2007. Andras Kovacs (andras@csevego.net)
 *                 Tamas Sule    (suletom@linux.csevego.net)
 * Portions Copyright (c) Russ Dill <Russ.Dill@asu.edu>
 *
 * Powered by NetClub, Gyor (http://internet.sth.sze.hu)
 *
 * This library is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation; either
 * version 2.1 of the License, or (at your option) any later version.
 *
 * This library is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with this library; if not, write to the Free Software
 * Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301, USA
 *
 *                     Modified by: Pedro Costa
 */

#include <stdio.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <time.h>
#include <string.h>
#include <sys/ioctl.h>
#include <net/if.h>
#include <errno.h>
#include <netinet/udp.h>
#include <netinet/ip.h>
#include <linux/if_ether.h>
#include <linux/if_packet.h>


#define timeout 2          // wait for 2 seconds

// Do not change something below.

#define VERSION "1.0b"
#define mac_bcast "\xff\xff\xff\xff\xff\xff"
int nagios_state=0;
struct dhcpMessage {
       u_int8_t op;
       u_int8_t htype;
       u_int8_t hlen;
```

```
        u_int8_t hops;
        u_int32_t xid;
        u_int16_t secs;
        u_int16_t flags;
        u_int32_t ciaddr;
        u_int32_t yiaddr;
        u_int32_t siaddr;
        u_int32_t giaddr;
        u_int8_t chaddr[16];
        u_int8_t sname[64];
        u_int8_t file[128];
        u_int32_t cookie;
        u_int8_t options[308]; /* 312 - cookie */
};

struct udp_dhcp_packet {
        struct ethhdr eth;
         struct iphdr ip;
         struct udphdr udp;
         struct dhcpMessage data;
} __attribute__ ((packed)) udp_dhcp_packet;

struct udp_dhcp_packet_for_checksum {
         struct iphdr ip;
         struct udphdr udp;
         struct dhcpMessage data;
};

u_int16_t checksum(void *addr, int count)
{
        /* Compute Internet Checksum for "count" bytes
         *         beginning at location "addr".
         */
        register int32_t sum = 0;
        u_int16_t *source = (u_int16_t *) addr;

        while (count > 1)  {
                /*  This is the inner loop */
                sum += *source++;
                count -= 2;
        }

        /*  Add left-over byte, if any */
        if (count > 0) {
                /* Make sure that the left-over byte is added correctly both
                 * with little and big endian hosts */
                u_int16_t tmp = 0;
                *(unsigned char *) (&tmp) = * (unsigned char *) source;
                sum += tmp;
        }
        /*  Fold 32-bit sum to 16 bits */
        while (sum >> 16)
                sum = (sum & 0xffff) + (sum >> 16);

        return ~sum;
```

```
}

int initialize_socket(struct sockaddr_ll * dest, unsigned char *dest_arp, int * fd,
unsigned char * mac,
                    char * ifname) {

    struct ifreq ifr;
    int ifindex;

    ifr.ifr_addr.sa_family = AF_INET;
    strcpy(ifr.ifr_name, ifname);

    memset(&(*dest), 0, sizeof(dest));

    if ((*fd = socket(PF_PACKET, SOCK_RAW, htons(ETH_P_IP))) < 0) {
       printf("Socket error: %s\n", strerror(errno));
       return -1;
    }

    if (ioctl(*fd, SIOCGIFINDEX, &ifr) == 0) {
       //printf("Adapter index: %d\n", ifr.ifr_ifindex);
       ifindex = ifr.ifr_ifindex;
    }
    else {
       printf("SIOCGIFINDEX failed: %s\n", strerror(errno));
       return -1;
    }
    if (ioctl(*fd, SIOCGIFHWADDR, &ifr) == 0) {
       memcpy(mac, ifr.ifr_hwaddr.sa_data, 6);
/* ************************************************* */
       //printf("Adapter hardware address: %02x:%02x:%02x:%02x:%02x:%02x\n",
       //     (unsigned int) mac[0], (unsigned int)mac[1], (unsigned int)mac[2],
(unsigned int)mac[3], (unsigned int)mac[4], (unsigned int)mac[5]);
/* ************************************************* */
} else {
       printf("SIOCGIFHWADDR failed: %s\n", strerror(errno));
       return -1;
    }

    dest->sll_family = AF_PACKET;
    dest->sll_protocol = htons(ETH_P_IP);
    dest->sll_ifindex = ifindex;
    dest->sll_halen = 6;

    memcpy(dest->sll_addr, dest_arp, 6);

    if (bind(*fd, (struct sockaddr *) dest, sizeof(struct sockaddr_ll)) < 0) {
       printf("Bind error: %s\n", strerror(errno));
       close(*fd);
       return -1;
    }
    return 0;
}
```

```
int raw_packet(struct dhcpMessage *payload, unsigned char *dest_arp, struct
sockaddr_ll * dest, int * fd,
               char * mac)
{
       int result;
       struct udp_dhcp_packet packet;
       struct udp_dhcp_packet_for_checksum packet_c;

       memset(&packet, 0, sizeof(packet));
       memset(&packet_c, 0, sizeof(packet_c));
       memcpy(&packet.eth.h_dest, (unsigned char *) mac_bcast, 6); /* broadcast */
       memcpy(&packet.eth.h_source, (unsigned char *) mac, 6);
       memcpy(&packet.eth.h_proto, (unsigned char *) "\x08\x00", 2); /* Type 0x0800
*/
       packet.ip.protocol = IPPROTO_UDP;
       packet.ip.saddr = INADDR_ANY; /* 0.0.0.0 */
       packet.ip.daddr = INADDR_BROADCAST; /* 255.255.255.255 */
       packet.udp.source = htons(68); /* source port */
       packet.udp.dest = htons(67); /* destination port */
       packet.udp.len = htons(sizeof(packet.udp) + sizeof(struct dhcpMessage)); /*
cheat on the psuedo-header */
       packet.ip.tot_len = packet.udp.len;

       memcpy(&(packet.data), payload, sizeof(struct dhcpMessage));
       memcpy(&(packet_c.data), payload, sizeof(struct dhcpMessage));

       packet_c.ip = packet.ip;
       packet_c.udp = packet.udp;
       packet.udp.check = checksum(&packet_c, sizeof(struct
udp_dhcp_packet_for_checksum));
       packet.ip.tot_len = htons(sizeof(struct udp_dhcp_packet) - sizeof(struct
ethhdr));
       packet.ip.ihl = sizeof(packet.ip) >> 2;
       packet.ip.version = IPVERSION;
       packet.ip.ttl = IPDEFTTL;
       packet.ip.check = checksum(&(packet.ip), sizeof(packet.ip));

       result = sendto(*fd, &packet, sizeof(struct udp_dhcp_packet), 0, (struct
sockaddr *) dest, sizeof(*dest));
       if (result <= 0) printf("Write error: %s\n", strerror(errno));
       return result;
}

static void init_packet(struct dhcpMessage *packet, char * mac)
{
        memset(packet, 0, sizeof(struct dhcpMessage));
        packet->op = 1;                          /* Bootrequest */
        packet->htype = 1;                  /* 10mb */
        packet->hlen = 6;                   /* 10mb_len */
        packet->cookie = htonl(0x63825363);        /* magic cookie */
       /* we're sending Discover */
       packet->options[0] = 0x35;
       packet->options[1] = 0x1;
       packet->options[2] = 0x1;
       /* end */
```

```
        packet->options[3] = 0xFF;          /* dhcp end delimiter */
       memcpy(packet->chaddr, mac, 6);
}

int send_discover(struct sockaddr_ll * dest, int * fd, unsigned char * mac)
{
       struct dhcpMessage packet;

       init_packet(&packet, mac);
       packet.xid = 0x3903F326;
       return raw_packet(&packet, (unsigned char *) mac_bcast, dest, fd, mac);
}

int get_raw_packet(struct dhcpMessage *payload, int fd)
{
       int bytes;
       struct udp_dhcp_packet packet;
       struct udp_dhcp_packet_for_checksum packet_c;
       u_int32_t source, dest;
       u_int16_t check;

       memset(&packet, 0, sizeof(struct udp_dhcp_packet));
       bytes = read(fd, &packet, sizeof(struct udp_dhcp_packet));
       if (bytes < 0)       return -1;  /* cannot read on socket */

       if (bytes < (int) (sizeof(struct iphdr) + sizeof(struct udphdr))) return -2;
/* msg too short */


       if (bytes < ntohs(packet.ip.tot_len)) return -2; /* truncated */

       /* ignore any extra garbage bytes */
       bytes = ntohs(packet.ip.tot_len);

       /* Make sure its the right packet for us, and that it passes sanity checks */
       if (packet.ip.protocol != IPPROTO_UDP || packet.ip.version != IPVERSION ||
           packet.ip.ihl != sizeof(packet.ip) >> 2 || packet.udp.dest != htons(68)
/* Client port */ ||
           bytes > (int) sizeof(struct udp_dhcp_packet) ||
           ntohs(packet.udp.len) != (short) (bytes - sizeof(packet.ip))) return -2;

       memcpy(payload, &(packet.data), bytes - (sizeof(packet.ip) +
sizeof(packet.udp) + sizeof(packet.eth)));

/* ************************************************* */
// Modified by: Pedro Costa

       if (ntohl(payload->cookie) != 0x63825363 /* Dhcp magic cookie */) return -2;

       printf("Dhcp server detected: mac=%02x:%02x:%02x:%02x:%02x:%02x, ",
packet.eth.h_source[0],
              packet.eth.h_source[1],packet.eth.h_source[2], packet.eth.h_source[3],
packet.eth.h_source[4],
              packet.eth.h_source[5]);
```

```
       printf("ip=%d.%d.%d.%d\n", packet.ip.saddr & 0xFF, packet.ip.saddr >> 8 &
0xFF,
              packet.ip.saddr >> 16 & 0xFF, packet.ip.saddr >> 24 & 0xFF);

       if (packet.eth.h_source[0] == 0x00 && packet.eth.h_source[1] == 0x25 &&
packet.eth.h_source[2] == 0x9c && packet.eth.h_source[3] == 0x8a  &&
       packet.eth.h_source[4] == 0xac && packet.eth.h_source[5] == 0x46) {
       printf("certo\n");
       nagios_state=0;}

       else{
       printf("errado\n");
       nagios_state=nagios_state+1;}
       return bytes - (sizeof(packet.ip) + sizeof(packet.udp) + sizeof(packet.udp));

}

/* ************************************************* */

int main (int argc, char * argv[]) {
    int i, fd;
    struct sockaddr_ll dest;
    struct dhcpMessage packet;
    clock_t start, end;
    struct tm * timeinfo;
    unsigned char mac[6];

    if (initialize_socket(&dest, (unsigned char *) mac_bcast, &fd, &(*mac), "eth0")
== -1);// return -1;
    send_discover(&dest, &fd, mac);

    start = clock();

    /* Set non-blocking mode for listening */
    int on = 1;
    ioctl(fd, FIONBIO, &on);

    while (1) {
/* ************************************************* */
// Modified by: Pedro Costa

       end = clock();
       if (((int)timeout) <= (int)(end/1000000))
       {
              printf("%d \n",nagios_state);
              if (nagios_state>0) return (2);// critical
              if (nagios_state==0) return (0);// return para o nagios OK
              if (nagios_state<0) return (2);

       }
       get_raw_packet(&packet, fd);

/* ************************************************* */
    }
}
```