Universidade do Minho

Gestão de Redes

Trabalho Prático Nº 2

Ferramenta SNMP para monitorização IP

Bruno Arieira
a70565

José Dias
a78494

February 9, 2019

Resumo

Este é um relatório de um trabalho prático que visa a implementação e o desenvolvimento de uma ferramenta **SNMP** que dispõe de funcionalidades capazes de permitir a monitorização IP, onde os resultados são mostrados numa interface *web*, por forma a facilitar a sua visualização a utilizadores como gestores de redes.

Índice

Índice de Imagens

Acrónimos

SNMP Simple Network Management Protocol 5

IP Internet Protocol 5

NMS Network-Management Systems 6

MIB Management Information Base 6

API Application programming interface 5

1 Introdução

Neste trabalho prático, temos como principal objetivo aplicar o conhecimento adquirido nas aulas lecionadas de Gestão de Redes, relativamente á matéria lecionada sobre o protocolo Simple Network Management Protocol (SNMP) com principal objetivo da criação de uma ferramenta SNMP que permita a monitorização IP.

Simple Network Management Protocol (SNMP) é um protocolo criado para facilitar a gestão e monitorização de dispositivos em redes Internet Protocol (IP). Estes dispositivos podem ser routers, computadores, servidores, etc. Tal como já foi referenciado, para este trabalho temos como principal foco a monitorização IP, através de uma ferramenta SNMP, que possibilite a um gestor de rede monitorizar a evolução dos valores calculados, relativamente ao número de octetos que saiem e entram nas interfaces de rede operacionais de um sistema ou *host* da rede local. Estes resultados são apresentados numa interface *web HTML* por forma a facilitar a sua visualização. Foi previamente necessária a instalação de um agente **SNMPv2c** e o pacote de desenvolvimento da linguagem de programação utilizada (**JAVA**) que disponibiliza as Application programming interface (API) precisas.

Em suma, este relatório está dividido de maneira a ser percetível a sua implementação e de forma a mostrar como foi encadeado o seu desenvolvimento. Assim, começamos por consolidar os principais conceitos teóricos necessários para a compreensão do projeto, de seguida passamos por demonstrar a solução desencadeada, depois mostramos a sua implementação explicando cada classe desenvolvida, posteriormente seguimos para os resultados, onde tratamos de exemplificar os passos necessários para a utilização da ferramenta criada e por fim terminamos com uma pequena conclusão, abordando a forma como o trabalho foi decorrendo.

2 Conceitos Teóricos

Simple Network Management Protocol (SNMP) é um protocolo que permite aos administradores de rede realizarem a gestão dos equipamentos de rede e diagnosticar os seus problemas[1]. Uma rede gerida pelo protocolo SNMP é formada por três componentes chaves:

- Dispositivos geridos;
- Agente;
- Network-Management Systems (NMS);

Os agentes são entidades que se encontram em cada interface, estes conectam-se aos dispositivos geridos na rede e permitem recuperar informações sobre os diversos objetos.

Um Network-Management Systems (NMS) é uma aplicação ou um conjunto de aplicações, que permite que os administradores de rede façam a gestão dos componentes independentes de uma rede, dentro de uma estrutura de gestão, de uma rede maior.

Switchs, *hubs*, *routers* e servidores são exemplos de Dispositivos geridos e que contêm objetos que podem ser geridos. Estes objetos podem ser informações materiais, parâmetros de configuração, estatísticas de desempenho e outros diretamente ligados ao comportamento atual do equipamento em questão. Estes objetos estão classificados numa espécie de uma base de dados chamado Management Information Base (MIB), que são conjuntos de dados organizados hierarquicamente usados para a gestão de entidades numa rede de comunicação.. O SNMP permite o diálogo entre gestores de rede e os agentes a fim de recolher os objetos desejados na MIB.

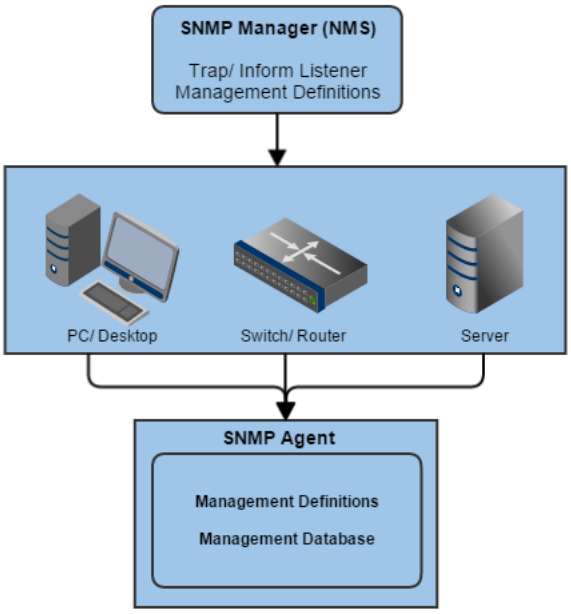

Figure 1: Imagem exemplificativa do SNMP

3 Solução

Antes de proceder ao devido desenvolvimento da ferramenta, foi necessário primeiramente estipular de forma objetiva as estatísticas assim como os parâmetros que irão ser abordados e a forma da qual iremos definir o polling. Com isto, foi preciso pesquisar de que maneira é que íamos consultar os devidos parâmetros, chegando á conclusão que os parâmetros que iam ser utilizados para objetivos de análise (*InOctets* e *OutOctets*) se encontram nas tabelas **IfInOctets** e **IfOutOctets**, sendo a consulta feita em torno desta tabela.

Sendo assim, percebemos que os valores dos octetos que entram, são os que mantém um *host* de uma determinada rede ocupado, e os octetos que saiem são os dados ou informação de resposta.

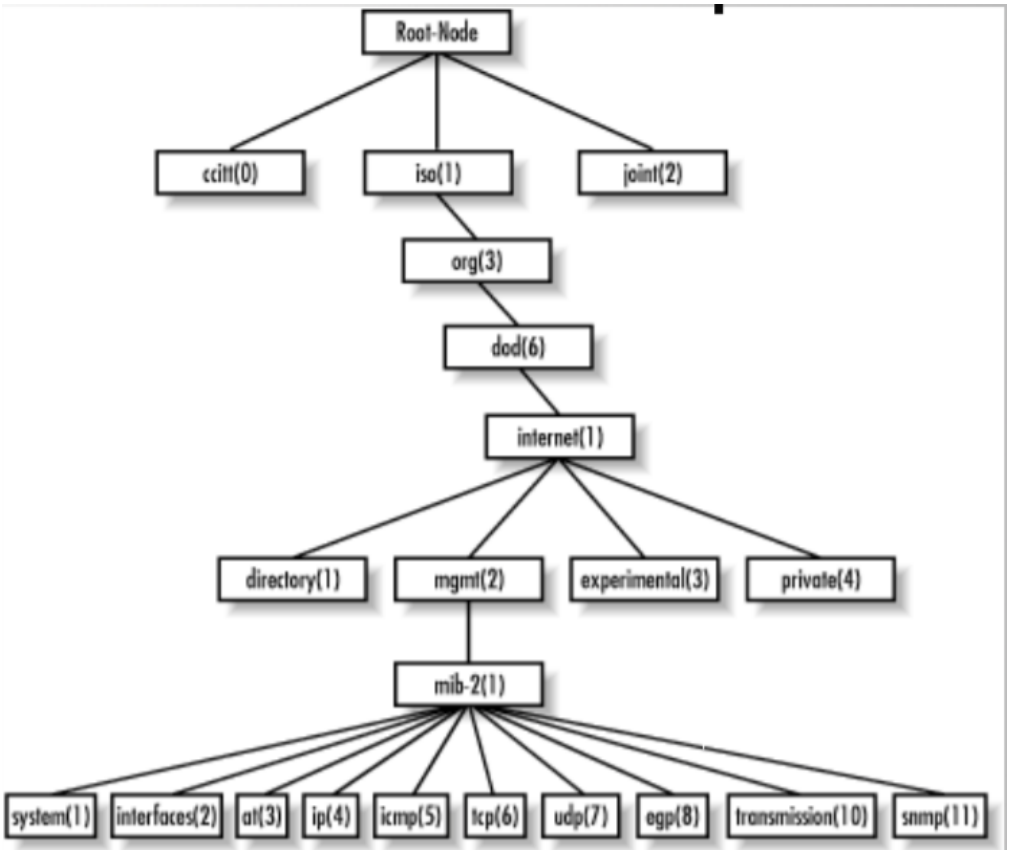

Figure 2: MIB relativa ao SNMP

- Através da informação relativa á MIB guardamos todas as interfaces:

  ".1.3.6.1.2.1.2.2"

- De seguida, para cada interface, guardamos os valores dos octetos de entrada e de saída, bem como a sua descrição, através do *object* **ifTable**:

  ifPhysAddress: ".1.3.6.1.2.1.2.2.6"

  ifInOctets: ".1.3.6.1.2.1.2.2.10"

  ifOutOctets: ".1.3.6.1.2.1.2.2.16"

Desta maneira, torna-se facilitado o processo de fazer a diferença dos octetos de entrada e saída para cada interface, sendo agora necessário estipular um *polling* para cada.

O *polling* definimos de acordo com a diferença dos octetos, em que caso o valor anterior da diferença dos octetos com a subtração pelo valor da diferença atual fosse maior que 50, diminuímos uma unidade ao *polling* e caso a subtração dos dois valores fosse menor que 10, aumentamos uma unidade.

4 Implementação

A estrutura do trabalho onde a ferramenta foi desenvolvida consiste em três ficheiros, de modo a permitir ao utilizador definir o seu endereço IP, a porta **UDP** e consultar como a diferença de octetos/bytes vai evoluindo ao longo do tempo, entre intervalos de polling estipulados. A classe *Par*, a classe *SNMPClient*, a classe *ifStatus* e a classe *Ux*(*user experience*), todas escritas em linguagem **JAVA**.

4.1 Classe SNMPClient

Esta classe foi criada de modo a guardar os dados necessários por forma a facilitar a implementação da interface e á consequente geração dos resultados. Primeiramente definimos o contrutor vazio, onde definimos o endereço e porta default (127.0.0.1:161) e depois a função *SNMPClient* que recebe um endereço para o caso do utilizador quiser definir um outro endereço **IP**. De seguida, guardamos todas as interfaces num *HashMap* (*ifList*) onde a key é o *mac address* que está associado a um objeto *ifStatus* e outro *HashMap* (*ifTrafficLog*) em que a cada key (*mac address*) está associado um objeto *Par*. A função *killAll* interrompe todas as threads e configura o Cliente com uma nova porta e endereço. Para começar a monitorização (*startMonitoring*), ao *HashMap* com as interfaces guardadas, atribui uma *thread* a cada. Elaboramos também uma função (*getTraffic*) cuja funcionalidade é guardar os octetos de entrada e os octetos de saída, que para cada interface é guardado um par destes valores, no objeto da classe *Par*. Posteriormente, a cada objeto *Par* criado, definimos um conjunto máximo relevante de gets a cada interfaces de cada vez a serem mostrados. Para algumas funções relativas á implementação de inicializar a ferramenta com um cliente recorremos a um blog introdutório da API de **SNMP4j** [2].

4.2 Classe Par

Nesta classe são definidos três *doubles*, onde serão guardados os octetos de entrada, saída e diferença para facilitar o modo de armazenar e aceder a estes valores.

4.3 Classe ifStatus

Em relação a esta classe, guardamos os dados relevantes relativos a cada interface como o *polling*. *mac address* e o respetivo índice. Relativamente ao *polling* (definido inicialmente como 5 segundos), caso a diferença atual dos octetos com a anterior seja maior que 50 diminui uma unidade, caso seja inferior que 10 aumenta uma unidade, tal como já foi referido anteriormente.

4.4 Classe Ux

Esta classe (*User Experience*) é a usada para dar apoio á nossa interface gerada assim como os resultados obtidos. Para a primeira opção, invocamos a função *start()* e *fillIfTable()* que inicia o SNMP com um cliente e guarda todas as interfaces. Para a segunda função, iniciamos a monitorização, associando cada *thread* a cada interface. Caso o cliente deseje configurar o endereço (3ªopção), todas as *threads* são interrompidas e feita outra procura de interfaces com a nova configuração. Relativamente á opção 4, através da função *printInterfaces()* os dados relativos aos octetos de entrada e de saída assim como a sua diferença e o respetivo *mac address*, são guardados numa tabela gerada em **HTML**. Na opção 0, é gerado automaticamente a interface gráfica web com os resultados, independentemente se o utilizador iniciou a monitorização ou nao. Para a implementação do gráfico, decidimos ocultar as interfaces que não continham *mac address* pois representa a máquina de onde está a ser feita a monitorização, sendo os octetos de entrada iguais aos de saída, não apresentando nenhuma diferença.

5 Resultados

Neste capítulo iremos fazer uma demonstração do modo de funcionamento da ferramenta elaborada. Para executar este trabalho é necessário que a versão instalada do java (**JDK**) seja igual ou superior á versão 10.

1ªPasso: Inicialmente teremos de executar o trabalho prático, onde em primeiro lugar é necessário encontrar as interfaces e posteriormente começar a monitorização.

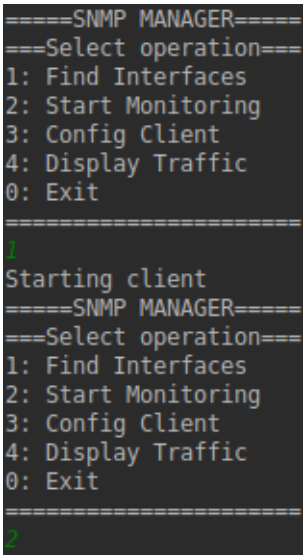

Figure 3: Menu Inicial

2ªPasso: De seguida, caso o endereço **IP** seja o default (127.0.0.1) e a porta **UDP** pretendida seja 161, não é necessário configurar cliente. Caso contrário, é necessário configurar com os dados prentendidos.

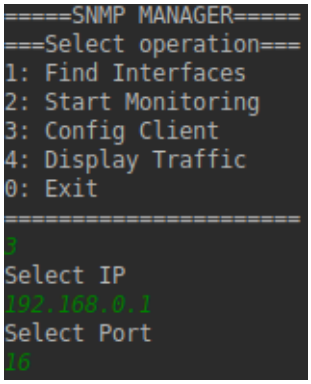

Figure 4: Configuração da porta **UDP** e endereço **IP**

3ªPasso: E por fim, seleciona-se a opção "Display Traffic" que calcula uma tabela com o *mac address* da interface, os octetos de entrada e saída, a sua diferença e um gráfico que mostra de que forma é que a diferença dos octetos vai mudando ao longo do tempo, e apenas quando sair ("*Exit*") o browser aparece mostrando os resultados.

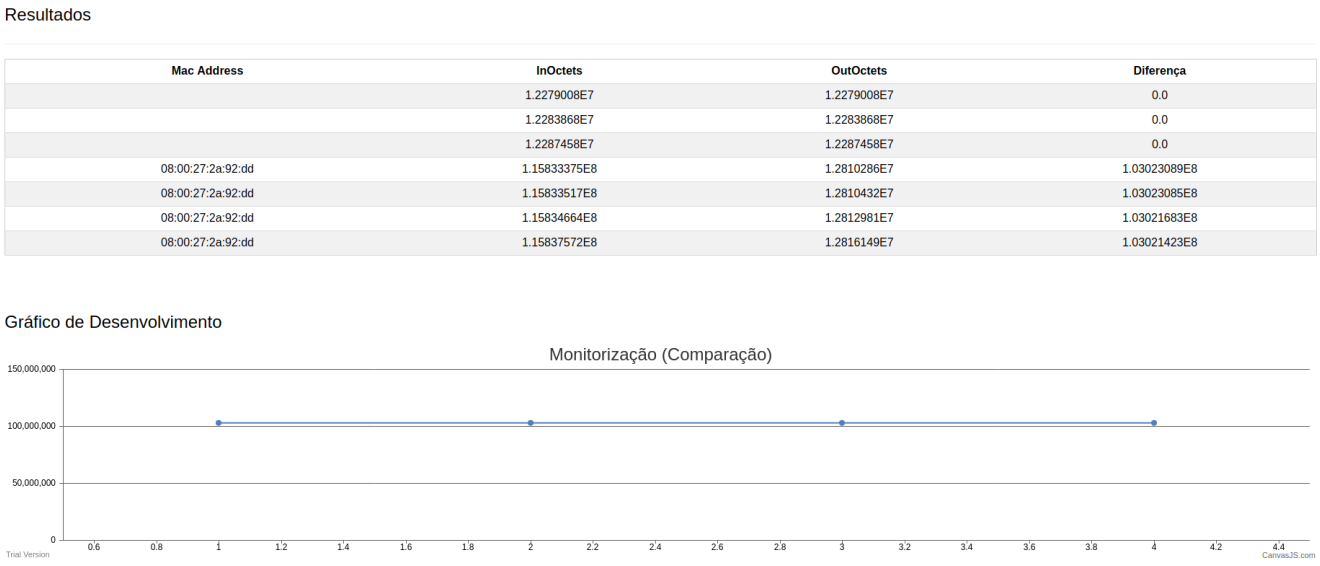

Figure 5: Resultado da Interface *Web*

[NOTA IMPORTANTE] Ao escolher a opção "*Start Monitoring*", deve-se esperar um determinado tempo antes de proceder ao "*Display Traffic*" por forma a carregar um conjunto relevante de gets a cada interface e fazer a respetiva monitorização.

6 Conclusão

Com este trabalho consolidamos os conceitos aprendidos das aulas práticas e teóricas da melhor forma, pois permitiu nos obter maior experiência com interacão do protocolo, assim como a utilização das APIs do **SNMP4j** de forma a possibilitar o desenvolvimento desta ferramenta relevante.

Tendo em conta o enunciado proposto, obtivemos algumas adversidades, algumas das quais passaram pela implementação da interface gráfica, assim como a imposição de *threads* para cada interface de forma a gerar o resultado final. No entanto, tudo que nos foi proposto pensamos que foi resolvido e implementado por forma a possibilitar uma adequada visualização dos resultados.

Concluindo, achámos um trabalho interessante, pois mais uma vez, ofereceu-nos bastante experiência com este protocolo relevante no âmbito da monitorização e gestão de redes, o que nos proporciona uma bagagem aumentada em termos profissionais.

References

[1] Simple Network Management Protocol (SNMP). https://searchnetworking.techtarget.com/definition/SNMP. [Online; accessed January 2018].

[2] Johan Rask. Introduction to snmp4j. https://blog.jayway.com/2010/05/21/introduction-to-snmp4j/, 2010. [Online; accessed 10-January-2019].