Universidade do Minho

Gestão de Redes

Trabalho Prático Nº 2

Servidor e criação e monitorização de containers com interface SNMP

Bruno Arieira

a70565

José Dias

a78494

February 9, 2019

Resumo

Este é um relatório do trabalho prático que visa o desenvolvimento de um agente **SNMP** que seja um servidor que permita a criação e gestão de containers. Para tal, inicialmente será necessário definir uma **MIB** e posteriormente deverá ser executado e contruído o agente que implemente o serviço num agente **SNMPv2c**. Este tipo de servido poderá ser usado por administradores.

Índice

Índice de Imagens

Acrónimos

SNMP Simple Network Management Protocol 5

IP Internet Protocol 5

MIB Management Information Base 6

API Application programming interface 5

OID Object Identifier 6

1 Introdução

O presente relatório diretamente relacionado com o trabalho desenvolvido, realizado no âmbito da unidade curricular Gestão de Redes, destina-se a apresentar uma solução coerente relativamente ao pedido no enunciado com principal objetivo de cimentar todos os conceitos abordados nas aulas laboratoriais e teóricas.

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. O principal foco deste trabalho prático é o desenvolvimento e implementação de um agente SNMP que seja um servidor que disponibilize ao administrador a funcionalidade de criação de *containers* a partir de um conjunto de imagens já determinadas assim como consultar o estado dos *containers* existentes utilizando SNMP. Para tal, foi necessária a criação definida de acordo com o enunciado do projeto. Foi previamente necessária instalação de um **agenteSNMPv2c** e o pacote de desenvolvimento da linguagem de programação utilizada (JAVA) que disponibiliza as Application programming interface (API) precisas, que neste caso foi a **SNMP4j**.

Resumindo, a estrutura deste relatório está dividida de maneira a ser percetível a sua implementação e de forma a mostrar como foi encadeado o seu desenvolvimento. Deste modo, iniciamos por abordar os principais conceitos teóricos necessários para a compreensão do projeto, de seguida passamos por demonstrar a solução desencadeada, depois mostramos a sua implementação explicando por cada etapa desenvolvida, posteriormente seguimos para os resultados e por fim terminamos com uma pequena conclusão, referindo de que maneira o trabalho foi decorrendo.

2 Conceitos Teóricos

Os conceitos teóricos que ajudam a perceber o âmbito deste trabalho prático relativos ao SNMP já foram todos relatados no último trabalho prático desenvolvido. Para não se tornar repetitivo, decidimos, neste relatório, acrescentar conceitos que ainda não foram abordados que são essenciais para este projeto.

A hierarquia Management Information Base (MIB) pode ser entendida como uma árvore de raiz da qual não se tem conhecimento, tendo seus ramos distribuídos por diferentes organizações. A camada mais alta de Object Identifier (OID) da MIB se encontra padronizada em diferentes organizações, enquanto que a camada mais baixa é alocada em organizações associadas. Essa arquitetura permite a gestão em todas as camadas do modelo de referência **OSI**. Assim, com as MIBs podendo ser definidas para cada área específica de informação, é possível extender as aplicações em bases de dados, correio eletrónico entre outras. Para cada objeto, a cláusula da sintaxe define o formato do inteiro e o **MAX-ACESS** define se é possível o acesso em termos de leitura ou escrita, ou até nenhum-acesso.

Um objeto gerido é uma das inúmeras características específicas de um dispositivo gerido. Objetos geridos podem assumir uma ou mais instâncias dos objetos essencialmente variáveis, identificados pelos seus OIDs. Com isto, existem dois tipos de objetos geridos:

- Escalares: definem uma única instância para os objetos;
- Tabulares: definem várias instâncias para objetos que são agrupadas em tabelas MIB;

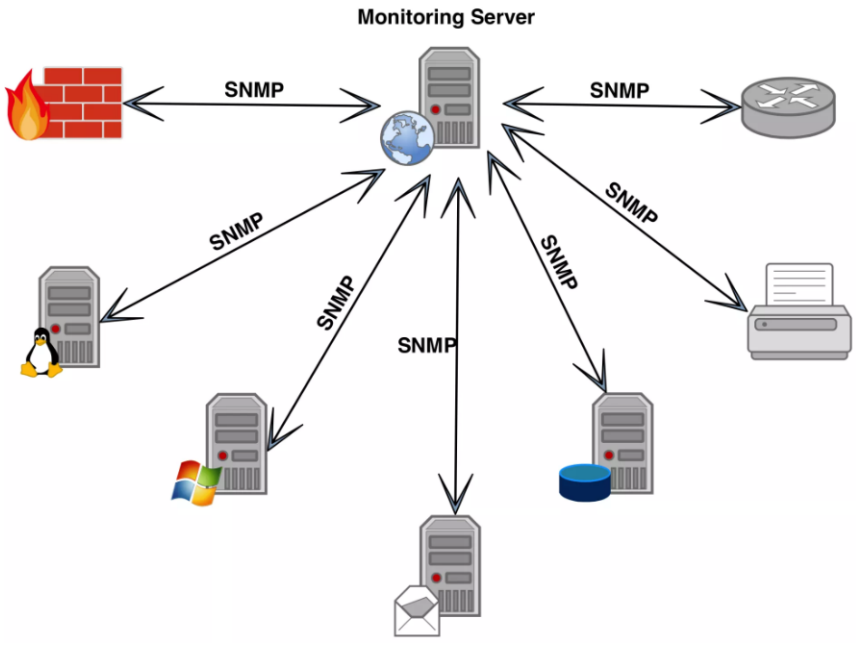

Figure 1: Aplicações SNMP

3 Solução

Neste capitulo, iremos abordar o método utilizado por forma a executar todas as fases contidas no enunciado deste trabalho prático.

A primeira tarefa realizada durante a execução deste projeto foi a definição da MIB. Para a implementação desta etapa, apenas nos decidimos guiar pelo enunciado, que continha o passos enumerados de como organizar e definir a nossa CONTAINERSHIP-MIB. O desenho relativo á MIB final é:

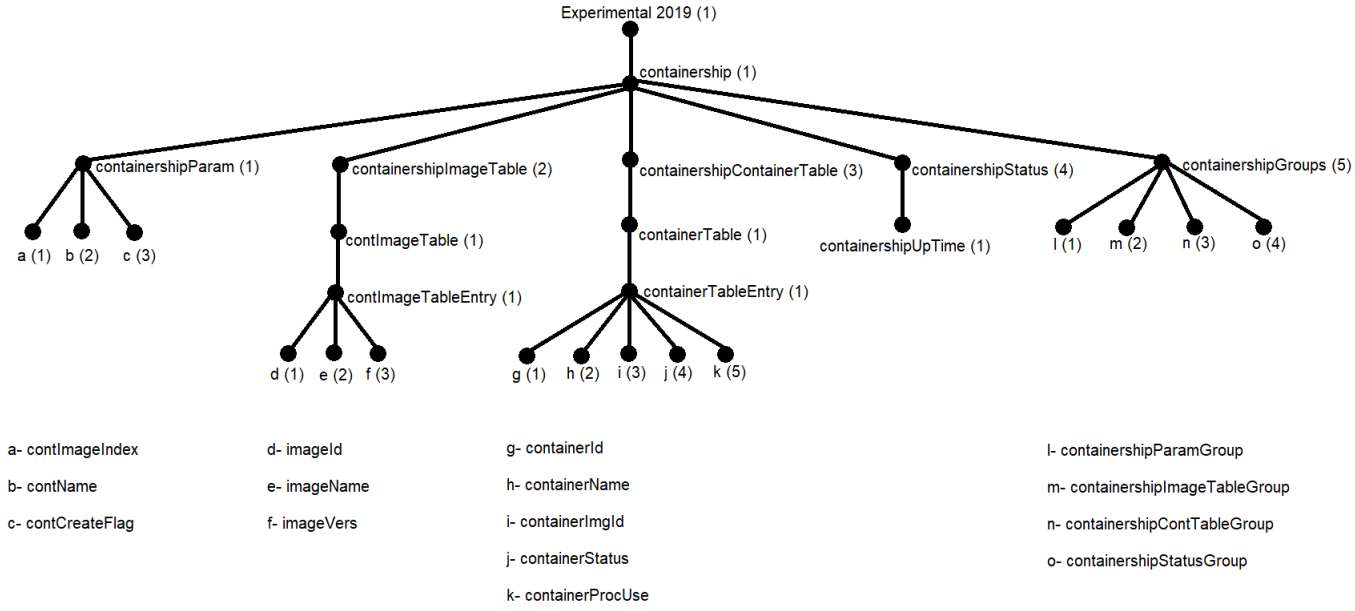

Figure 2: Desenho da MIB implementada

Posteriormente passamos á construção de um agente SNMP. Para a construção do agente recorremos á API do SNMP4J. Tomamos como base para a construção um projeto [1] encontrado na web que define a criação de um agente básico com recurso á API referida. Partindo do agente aí retratado definimos o nosso com os requisitos necessários.

Após a verificação do agente partimos para o registo dos objetos definidos na fase da construção da MIB. O registo foi efetuado recorrendo a um programa denominado **AgenPro**, que foi um dos catalisadores a efetuar algumas mudanças na nossa MIB inicial, foi aqui que criamos o grupo *ContainershipGroups* pois além de ser recomendado ter todos os objetos dentro de um `OBJECT-GROUP` estes eram obrigatórios para a execução do programa.

4 Implementação

Nesta secção iremos proceder á explicação relativamente a cada uma das etapas desenvolvidas deste trabalho prático, com objetivo de esclarecer a implementação do mesmo. Como já foi referido anteriormente para a definição da MIB recorremos aos dados do guião do projeto. A MIB inicialmente estava dividida em 4 subgrupos (os referidos no guião), mas na sua iteração final encontra-se divida em 5 subgrupos acrescentando aos 4 pré existentes 1 novo subgrupo contendo os `OBJECT-GROUP`.

Para a criação do agente referimos já a nossa base num outro projeto [1] e que recorremos ao programa **AgenPro** para efetuar o registo dos objetos da MIB. Aí verificamos que não era possível efetuar a operação `snmpset` e como tal foi necessário adicionar permissões, isto vemos na função `addViews` na classe `Agent`.

Após isto todos os objetos da MIB estavam registados no agente e com as permissões corretas, como tal foi apenas necessário preencher a MIB com os valores necessários e a partir daí passamos aos testes do agente onde verificamos alguns erros na definição do nosso agente.

5 Resultados

Neste capítulo iremos fazer uma demonstração dos resultados da implementação do trabalho prático. Para executar este trabalho é necessário que a versão instalada do java (**JDK**) seja igual ou superior á versão 10.

```
bruno@arieira:~$ snmpwalk -m +CONTAINERSHIPv2-MIB -v2c -c public localhost:6666
1.3.6.1.3
CONTAINERSHIPv2-MIB::contImageIndex.0 = INTEGER: 0
CONTAINERSHIPv2-MIB::contName.0 = ""
CONTAINERSHIPv2-MIB::contCreateFlag.0 = INTEGER: noCreate(0)
CONTAINERSHIPv2-MIB::imageId.1 = INTEGER: 1
CONTAINERSHIPv2-MIB::imageId.2 = INTEGER: 2
CONTAINERSHIPv2-MIB::imageId.3 = INTEGER: 3
CONTAINERSHIPv2-MIB::imageName.1 = STRING: "ubuntu"
CONTAINERSHIPv2-MIB::imageName.2 = STRING: "php"
CONTAINERSHIPv2-MIB::imageName.3 = STRING: "postgre"
CONTAINERSHIPv2-MIB::imageVers.1 = STRING: "latest"
CONTAINERSHIPv2-MIB::imageVers.2 = STRING: "7.2-apache"
CONTAINERSHIPv2-MIB::imageVers.3 = STRING: "9.5"
CONTAINERSHIPv2-MIB::containershipUpTime.0 = STRING: "08/02/2019 04:03:34"
```

Figure 3: Resultado do comando *walk* imediatamente após executar o agente

```
bruno@arieira:~$ snmptable -m +CONTAINERSHIPv2-MIB -v2c -c public localhost:6666 CONTAINERSHIPv2-MIB::contImageTable
SNMP table: CONTAINERSHIPv2-MIB::contImageTable

 imageId imageName    imageVers
       1  "ubuntu"     "latest"
       2     "php" "7.2-apache"
       3 "postgre"        "9.5"
```

Figure 4: Reconhecimento de tabelas por parte do agente

```
bruno@arieira:~$ snmpset -m +CONTAINERSHIPv2-MIB -v2c -c public localhost:6666 C
ONTAINERSHIPv2-MIB::contImageIndex.0 i 1
CONTAINERSHIPv2-MIB::contImageIndex.0 = INTEGER: 1
```

Figure 5: Aplicação do comando *set* ao índice

```
bruno@arieira:~$ snmpset -m +CONTAINERSHIPv2-MIB -v2c -c public localhost:6666 C
ONTAINERSHIPv2-MIB::contName.0 s "Container1"
CONTAINERSHIPv2-MIB::contName.0 = STRING: "Container1"
```

Figure 6: Aplicação do comando *set* ao nome

```
bruno@arieira:~$ snmpset -m +CONTAINERSHIPv2-MIB -v2c -c public localhost:6666 C
ONTAINERSHIPv2-MIB::contCreateFlag.0 i 1
CONTAINERSHIPv2-MIB::contCreateFlag.0 = INTEGER: create(1)
```

Figure 7: Aplicação do comando *set* á *flag*

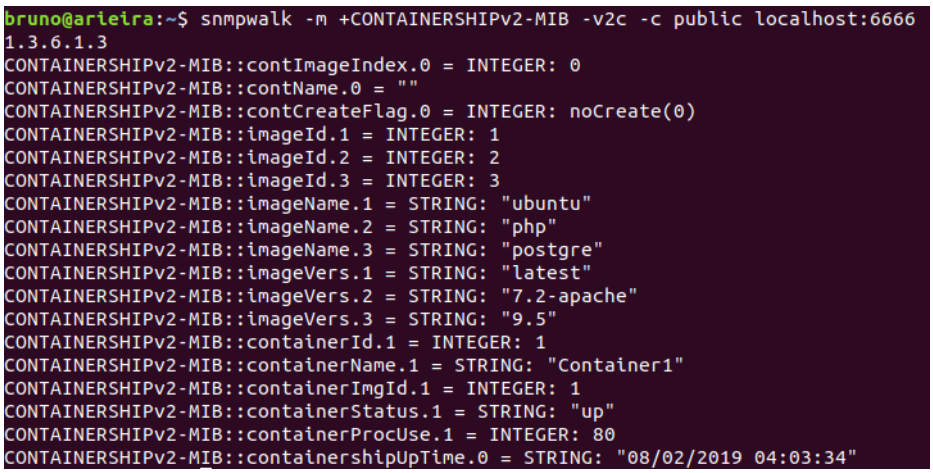

Figure 8: Resultado do comando *walk* imediatamente após efetuar *set* da flag a 1

6 Conclusão

Tal como já foi mencionado, o principal objetivo deste trabalho prático é o desenvolvimento e implementação de um agente **SNMP!** que seja um servidor que disponibilize ao administrador a funcionalidade de criação e consulta de containers a partir de um conjunto de imagens já determinadas.

Ao longo do desenvolvimento deste trabalho prático obtivemos algumas dificuldades relativamente á execução de todas as tarefas propostas no enunciado, nomeadamente as tarefas **E** e **G**, que não chegaram a ser implementadas. Na tarefa **E** (Implementação de mudanças de estado) não fomos capazes de a demonstrar durante a apresentação devido á má implementação do objeto da MIB relativo ao estado do *container*, mas já se encontra resolvido. Em relação á tarefa **F** (Implementação *containershipStatus*) apenas desenvolvemos o armazenamento da data do sistema quando o agente foi iniciado, sendo que as restantes tarefas foram realizadas.

Em síntese, não foi possível realizar com sucesso todas as funcionalidades dos requisitos pedidos no enunciado relativamente a este trabalho prático. Apesar das adversidades, achámos um trabalho bastante completo e interessante, pois proporcionou-nos um maior conhecimento em relação a esta vertente inserida em gestão de redes.

References

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