

Gestão de Redes

Ferramenta SNMP para Monitorização de Processos

Trabalho Prático 2

Hugo Machado - A80362

Docentes: Bruno Dias
Fábio Gonçalves

Índice

Introdução.....	3
Ferramenta desenvolvida.....	4
Primeiro Passo.....	4
Segundo Passo.....	5
Terceiro Passo.....	6
Quarto Passo.....	6
No Terminal.....	6
Na Web.....	7
Instruções de uso.....	10
Conclusão.....	11

Introdução

Neste trabalho pratico, pretende-se criar um programa de monitorização dos processos presentes numa estação de trabalho. Utilizando o **SNMP** como ferramenta para esse fim.

Os requisitos propostos foram os seguintes, o endereço e porta do agente fossem configuráveis e uma interface visual para facilitar a monitorização.

Ferramenta desenvolvida

A ferramenta desenvolvida, ou programa criado, foi feito em C. Apesar de não ser a linguagem mais indicada utilizar com *web* é uma linguagem leve, eficiente, *low-level* e muito utilizada no que toca a sistemas operativos e redes.

Para a concretização deste trabalho prático usei as bases dadas nas aulas de Gestão de Redes e de Sistema Operativos, para guardar a informação obtida de forma organizada fiz uso de estruturas e ficheiros guardar a informação obtida.

Primeiro Passo

Inicialmente para a criação desta ferramenta tive procurar quais os OID's a utilizar. Após algum estudo e consulta da HOST-RESOURCES-MIB, descobri que os OID's que iria necessitar são os seguintes:

- hrSystemProcesses*: Número de processos ativos na máquina;
- hrSWRunIndex*: Índice do processo, ou o número identificador do processo (PID);
- hrSWRunName*: Nome do programa executado pelo processo;
- hrSWRunPerfCPU*: Número de centéssegundos, consumidos do CPU pelo processo;
- hrSWRunPerfMem*: Número em *Kbytes* utilizados pelo processo.

Mais tarde, devido às unidades que o *hrSWRunPerfCPU* devolve, tive de efetuar um pedido para o *hrSystemUptime*, este também é devolvido em centéssegundos, e assim é possível calcular a percentagem deste valor.

Segundo Passo

De seguida, tive de decidir a forma como iria efetuar os pedidos e os argumentos necessários para cada pedido.

Depois de alguma investigação decidi que não iria utilizar as bibliotecas existentes do SNMP em C e iria usar *exec's*, estas são chamadas ao sistema que permitem executar comandos da *bash*. Com os *exec's* tive de fazer uso *pipe's anónimos*, *dup2's*, para que o resultado do pedido não fosse escrito no terminal e para que o pudesse guardar.

A função desenvolvida para cada pedido é a seguinte:

```
int exec(char *c, char *p1, char *p2, char *p3, char *p4){
    int status = -1, exitvalue=-1, p[2];
    pipe(p);
    if (exitvalue!=0) {
        if (fork() == 0) {
            dup2(p[1],1); close(p[0]);
            if (execlp(c, c, "-v2c",ipport,"-OU","-Os","-Ov","-Oe", p1, p2, p3, p4, NULL)<0) {
                perror("Abortar: Erro no exec\n");
                exit(-1);
            }
        }
        close(p[1]);
        wait(&status);
        exitvalue = WEXITSTATUS(status);
    }
    memset(result, 0, SIZE); readln(p[0], result, 1); close(p[0]);
    return exitvalue;
}
```

Descricao dos argumentos utilizados:

- c: esta variável contém o comando escolhido;
- ipport: esta variável contém o endereço *ip* da máquina e a porta escolhida;
- OU: não imprimir unidades;
- Os: imprimir o último valor simbólico *OID*;
- Ov: apenas imprimir os valores;
- Oe: imprimir enumerações numericamente;
- p(x): esta(s) variável(eis) contém os *OID's* escolhidos.

Terceiro Passo

Com o código perto do fim, descobri que o comando *snmpbulkget* devolve no máximo 100 respostas, o que me levou a contar o número de pedidos que fazia, e quando fazia os pedidos por índice, tive de ter o cuidado de estar a par de qual foi o índice do último pedido.

Acrescentei também um algoritmo(*bubble sort*) para me ordenar os valores de cada pedido.

Quarto Passo

Por fim, depois de me assegurar do funcionamento do programa e da sua robustez, adicionei duas funcionalidades, estas são para o funcionamento da ferramenta com a *web* e com o *terminal*, estes distinguem-se através de diferentes argumentos na execução do programa.

No Terminal

Os intervalos de tempos entre pedidos não são configuráveis, mas o programa por si faz os pedidos sem interrupções, ou seja sempre acaba de fazer todos os pedidos, de os guardar, organizar e imprimir ele volta ao mesmo processo num *loop* infinito. Os valores são impressos num formato visualmente legível e quando em modo de *terminal* para sair usa-se a combinação *ctrl^c*.

Exemplo:

```
[HORA] : [1]:[19]:[22]

PID      %CPU    %MEM     NAME
2766      6.9     4.6     "firefox"
27850     0.1     2.9     "soffice.bin"
2333      2.7     2.6     "gnome-shell"
1627      0.0     2.4     "gnome-shell"
27443     0.1     2.4     "Web Content"
562       0.1     2.4     "Web Content"
3046      0.6     1.9     "Web Content"
27978     0.0     1.8     "Web Content"
9163      0.1     1.8     "vlc"
2913      0.1     1.7     "WebExtensions"
2076      1.7     1.5     "Xorg"
30707     7.8     1.5     "Web Content"
26405     0.1     1.4     "myki"
31870     0.0     1.4     "code"
2830      0.1     1.4     "Web Content"
16674     0.1     1.2     "Mailspring defa"
29743     0.0     1.1     "Web Content"
20097     0.2     1.0     "Web Content"
31918     0.0     1.0     "code"
26458     0.0     1.0     "myki"

(ctrlt^c = To exit)
```

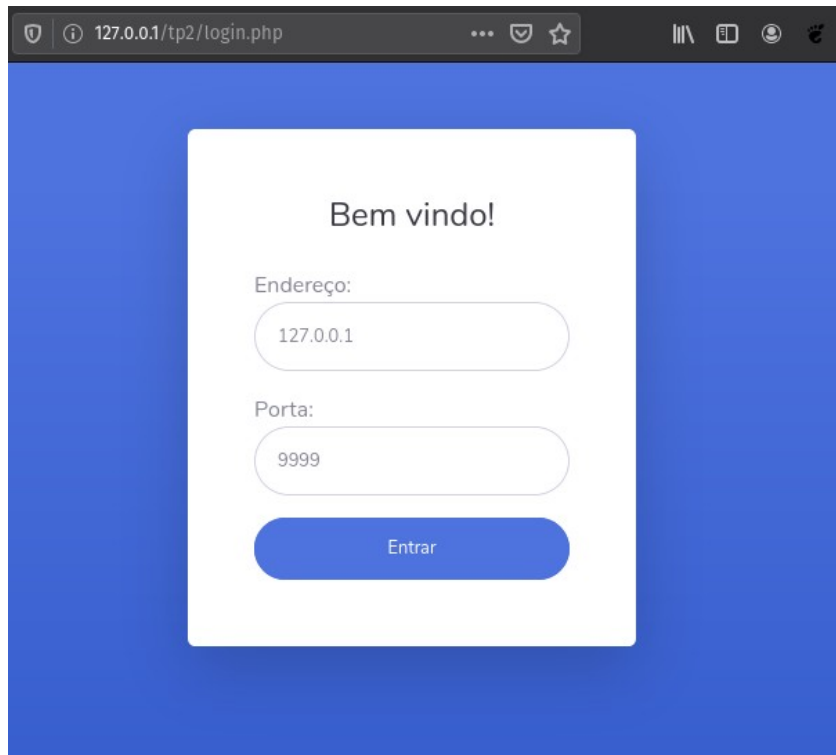
Na Web

Visto que iniciei o a ferramenta em *C*, a integração com a *Web* tornou-se um desafio, pois esta linguagem não foi feita para interligar com a *Web*. Mas para contornar o problema utilizei o comando *shell_exec()* de *PHP*, esta função permite-me correr um executável e guardar o seu resultado dentro de uma variável, desta forma consegui executar a minha ferramenta através do *website*.

Para a organizacao de dados do resultado do *shell_exec()* decidi na minha ferramenta, quando utilizada para *Web*, imprimir o resultado em formato *json* de assim consegui converte-la para múltiplos *array's* e utilizar livremente a informação no meu *website*.

Em relação aos intervalos de monitorização, na interface *web*, os pedidos são feitos a cada *refresh*.

Quando acedemos ao *website*, encontramos uma pagina que nos pedes os dados da maquina e porta do agente ao qual queremos aceder.



The image shows a web browser window with a dark address bar displaying the URL "127.0.0.1/tp2/login.php". The main content area has a solid blue background. Centered on this background is a white rectangular form. At the top of the form, the text "Bem vindo!" is displayed. Below this, there are two input fields. The first is labeled "Endereço:" and contains the text "127.0.0.1". The second is labeled "Porta:" and contains the text "9999". Below these two fields is a blue button with the white text "Entrar".

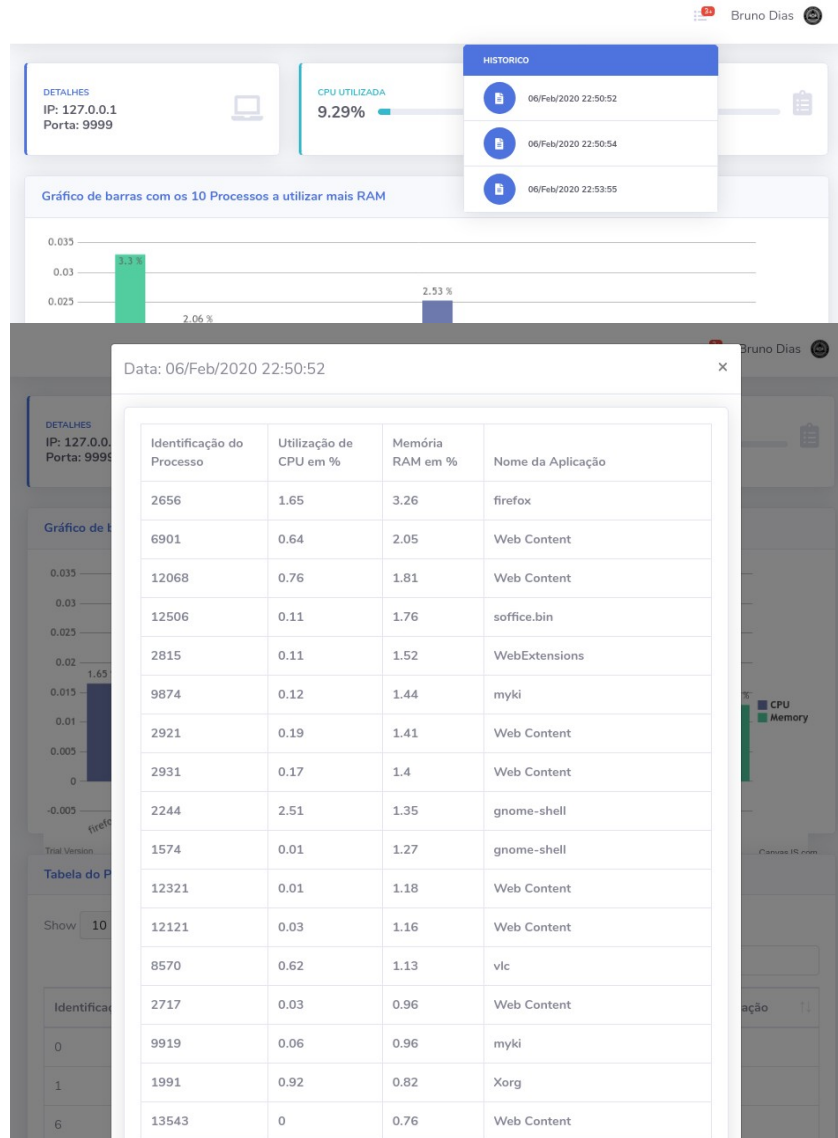
Na pagina seguinte, podemos consultar múltiplas informações da máquina, tais como percentagem de *CPU* e *RAM* utilizados, os 10 processos a utilizar mais *RAM* e em baixo uma tabela com a identificação, utilização de *CPU* e *RAM*, e nome da aplicação do respectivo processo.



Copyright © Gestão de Redes 2019 Universidade do Minho

Universidade do Minho
Mestrado Integrado em Engenharia de Telecomunicações e Informática
2019/202

Por fim, introduzi uma funcionalidade adicional, na minha ferramenta *C*, para poder ter um acesso a um histórico na página *web*, este consiste em fazer um ficheiro com todos os dados daquele momento. Depois a minha página tem um menu que nos faz a listagem de todos os momentos/ficheiros gravados, e permite-nos abrir para consultar a sua informação.



Instruções de uso

Para a utilização da ferramenta em modo de terminal, basta colocar o ficheiro *C* na diretoria desejada e escrever os seguinte comandos.

```
$make tp2
```

ou

```
$gcc -o tp2.c tp2
```

Para criar o executável.

```
$tp2 127.0.0.1 9999 1
```

Para executar.

Para a utilização na *web*, se estiver a utilizar o *apache2 webserver*, é necessário inicia-lo, e de seguida colocar a pasta com o conteúdo do *website* na diretoria */var/www/html/*. Depois através do *browser* é só aceder à pagina <http://127.0.0.1/tp2>.

Conclusão

Com a conclusão deste trabalho prático, ganhei maiores aptidões e interesse em relação à ferramenta e protocolo *SNMP*.

Por fim todos os objetivos foram alcançados com sucesso, tanto a interface *web* como a ferramenta de terminal, funcionando como pedido.