UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CMP193/INF01015 – Gerência e Aplicações em Redes
PROFESSORES: Juliano Wickboldt e Jéferson Campos Nobre

Lista de Exercícios 02
Aula 06 - Arquitetura de Gerenciamento da Internet (Aula prática)

***O exercício deve ser resolvido individualmente ou em dupla. A entrega deve ser realizada pelo Moodle, em um único documento (.pdf).**

Nessa aula vamos explorar algumas técnicas para desenvolvimento de sistemas baseados em SNMP usando uma linguagem de programação. No caso, vamos usar a biblioteca Easy SNMP em Python. Nessa aula vamos explorar apenas uma pequena parte das funcionalidades disponíveis nesta biblioteca. A documentação completa da biblioteca está disponível no site http://easysnmp.readthedocs.io/.

Instalando bibliotecas necessárias

Python já vem instalado por padrão no Linux, vamos precisar instalar apenas as outras bibliotecas para usar os comandos do SNMP. Comece instalando os requisitos:

```
# sudo apt install libsnmp-dev gcc python-dev python-pip python-setuptools snmp-mibs-downloader
```

Estou assumindo que você já tem instalado o Net-SNMP. Se não for o seu caso, instale também os pacotes descritos nas Instruções para Instalação do Net-SNMP no Linux. É necessário comentar a linha que define as MIBs a serem carregadas no snmp.conf.

Agora instale o Easy SNMP usando o pip (instalador de pacotes do Python):

```
# sudo pip install easysnmp
```

Primeiro programa escrito em Python com Easy SNMP

Vamos fazer um primeiro teste bastante simples com a biblioteca instalada. Crie um novo arquivo em Python com o seguinte conteúdo:

```
# Primeiro precisamos importar a biblioteca do Easy SNMP
from easysnmp import Session

# Depois podemos criar uma sessao (comando Session) SNMP para usar em todas as requisicoes (get, set, etc.) posteriores
session = Session(hostname='localhost', community='public', version=2)

# Agora vamos fazer um simples SNMP GET REQUEST pelo objeto sysName.0 colocando o valor retornado na variavel name
name = session.get('sysName.0')

# Agora basta imprimir a variavel name
print (name)
```

Note que você pode alterar o hostname='localhost' para outro nome ou IP de um agente SNMP que você deseja consultar.

Salve o arquivo com algum nome, por exemplo, easyget.py e execute ele com o seguinte comando:

```
# python easyget.py
```

A saída na tela do seu computador deve aparecer parecida com isso:

<SNMPVariable value='TESTE' (oid='sysName', oid_index='0', snmp_type='OCTETSTR')>

A primeira vista esse formato pode parecer um pouco estranho se você não está acostumado com Python. Resumidamente, esse formato indica que a variável **name** que usamos no nosso programa é um objeto em Python (não confunda com objetos do SNMP) do tipo SNMPVariable, sendo que ele possui alguns atributos interessantes para nós.

Principalmente, vamos precisar do valor (value) e talvez do tipo (snmp_type). Como fazer para obter essas informações? Ao invés de usar a variável name diretamente, podemos usar **name.value** para obter o valor ou **name.snmp_type** para obter o tipo.

Faça o teste, edite o seu programa e imprima na tela com o comando print do Python o valor e o tipo do objeto SNMP recuperado. Faça outros testes com outros objetos SNMP de MIBs que você conheça (contadores das interfaces de redes, por exemplo).

Exercício 1

Edite o seu primeiro programa acima para apresentar mais informações sobre o host do agente. Incluia na saída do programa pelo menos as seguintes informações:

1. Há quanto tempo o sistema está no ar (uptime)?
2. Qual o nome do host?
3. Quantos processos estão rodando no host?
4. Quantas interfaces de rede tem o host?
5. Quais os endereços físicos (MAC) das interfaces?
6. Quais os endereços IP associados ao hosts?

Um exemplo mais elaborado

Nesse exemplo a seguir vamos fazer primeiramente um SNMP GET para descobrir o número de interfaces do dispositivo gerenciado. Posteriormente, vamos usar um SNMP GET BULK para pegar os contadores de bytes in/out de todas as interfaces do dispositivo de uma só vez. Novamente os comentários devem servir para ajudar a entender o que cada linha faz.

```
# Primeiro precisamos importar a biblioteca do Easy SNMP
from easysnmp import Session

# Depois podemos criar uma sessao (session) SNMP para usar para todas as requisicoes
session = Session(hostname='localhost', community='public', version=2)

# Agora vamos fazer um SNMP GET REQUEST para descobrir o numero de interfaces do dispositivo gerenciado
ifNumber = session.get('ifNumber.0')

# Imprimir a quantidade de interfaces
print "Numero de interfaces", ifNumber.value

# Eh preciso converter o valor de ifNumber para inteiro porque o Easy SNMP nao faz isso por padrao
numInterfaces = int(ifNumber.value)

# Fazendo uma requisicao bulk com dois contadores in/out das interfaces de uma vez (repetindo conforme a quantidade de interfaces), isso vai retornar numInterfaces*2 objetos
countBulk = session.get_bulk(['ifInOctets', 'ifOutOctets'], 0, numInterfaces)

# Aqui vou definir duas listas (formalmente sao chamados de dicionarios) para armazenar os valores de contadores de cada interface
valifInOctets = {}
```

```
valifOutOctets = {}

# O nosso countBulk vai ter uma lista com numInterfaces*2 objetos, entao sera necessario repetir os comandos considerando a quantidade de interfaces (similar ao que foi feito no exemplo basico primeiro.py)
for i in range(numInterfaces):
    # Os contadores in/out da interface i estarao na lista um apos o outro (posicoes i e i+1)
    ifInOctets = countBulk[i]
    ifOutOctets = countBulk[i+1]

    # Convertendo os valores para inteiro
    valifInOctets[i] = int(ifInOctets.value)
    valifOutOctets[i] = int(ifOutOctets.value)

    print "Informacoes da Interface", i
    print "- In", valifInOctets[i], "(bytes)"
    print "- Out", valifOutOctets[i], "(bytes)"
```

Exercício 2

Agora vamos ao desafio...

Tente adaptar o programa acima para fazer o seguinte:

1. Faça um monitoramento dos contadores ifInOctets e ifOutOctets, assim como no programa exemplo, e depois faça o programa aguardar alguns segundos (5, por exemplo).
2. Logo em seguida faça outro monitoramento igual, recebendo novos valores para os contadores (você vai precisar definir novas variáveis para isso).
3. Ao invés de mostrar os contadores como estamos fazendo no nosso programa exemplo, mostre a taxa calculada pelo programa nesse intervalo de tempo.
4. Para calcular a taxa você vai precisar fazer algo como, **(contadorMonitoramento2-contadorMonitoramento1)/tempoDeEspera**.
5. Sempre que essa taxa ultrapassar um determinado limite, por exemplo, 5 megabytes por segundo, mostre uma mensagem de aviso na tela.

Mais umas dicas.

Utilize:

```
import time
```

E depois:

```
time.sleep(5)
```

No seu programa para fazer ele aguardar 5 segundos entre um monitoramento e outro.

Tutorial rápido para geração de tráfego

Para testar o programa feito na outra atividade prática dessa aula, vamos precisar gerar tráfego para ver os contadores mudando.

Seguem alguns exemplos de comandos no Linux que podem ser úteis para isso:

Ping

Ping é um comando que pode ser usado para enviar tráfego ICMP para praticamente qualquer outra

máquina sem precisar instalar nada.

```
# ping <IP_DA_OUTRA_MAQUINA>
```

Você pode também utilizar alguns parâmetros extras para gerar mais tráfego, por exemplo.

```
# ping -i 0.2 -s 1024 <IP_DA_OUTRA_MAQUINA>
```

-i define o intervalo entre os pacotes enviados

-s define o tamanho dos pacotes enviados

Utilize o comando **man ping** para ver todas as opções de parâmetros disponíveis nessa ferramenta.

Iperf3

O Iperf é uma ferramenta bem mais elaborada para gerar tráfego para teste de desempenho, mas essa você vai precisar instalar. Vamos fazer isso em duas máquinas, uma vai ser o servidor e outra o cliente. Pode combinar com algum colega para fazer os testes entre as VMs de vocês.

Instale o iperf3 com:

```
# sudo apt install iperf3
```

Na máquina que vai ser o **servidor**, digite:

```
# iperf3 -s
```

Na máquina que vai ser o cliente

```
# iperf3 -c <IP_DO_SERVIDOR>
```

Assim como fizemos para o ping, para ver mais informações de parâmetros que podemos utilizar, acesse o manual da ferramenta com o comando man iperf3.

Entregar

Um relatório simples com detalhes de como os experimentos foram executados, relatando quais objetos das MIBs foram usados para obter as informações dos sistemas, como vocês geraram tráfego para ativar as notificações, etc. Podem incluir prints das saídas dos programas escritos. Entreguem também os fontes em Python.