


IC - Instituto de Computação
UNICAMP - Universidade Estadual de Campinas
MC833 – Programação em Redes de Computadores
Primeiro Semestre 2024

Projeto 2: Analisando o tráfego de rede no Wireshark e no Mininet.

Você tem a tarefa de projetar e analisar uma rede composta por quatro *hosts* e um *switch* OpenFlow usando o Mininet. A tarefa envolve:

1. Configuração de rede (Mininet):

- Projetar uma topologia Mininet com:
 - Quatro *hosts* (por exemplo: h1, h2, h3, h4).
 - Um único *switch* habilitado para OpenFlow (por exemplo, s1).
- Para tanto, será necessário instalar o Mininet e para construir a topologia usar os seguintes comandos:

 O ambiente de simulação de redes foi **preparado previamente no laboratório**, e está pronto para ser utilizado. Para acessá-lo, siga os passos abaixo:

Abra o terminal da máquina local;
Digite o seguinte comando para iniciar a máquina virtual:

vm-mininet

A máquina virtual será executada automaticamente. Assim que o sistema carregar, **digite a senha** abaixo para fazer login:

mininet

- Para instalar nativamente a partir do código-fonte, primeiro você precisa obter o código-fonte com este comando:

git clone <https://github.com/mininet/mininet>

- Assim que tiver o código-fonte, o comando para instalar o Mininet é:
mininet/util/install.sh

- Para executar o Mininet:
sudo mn

- Para fazer uma topologia de 4 *hosts* e 1 *switch*:
sudo mn --topo single,4

2. Geração de Tráfego (Ping):

- Use os comandos ping ou pingall do Mininet para tráfego controlado:

- 200 pacotes de ping de h1 a h3
- 200 pacotes de ping de h2 a h4

- Para tanto, usar os seguintes comandos para enviar pacotes:

```
mininet > h1 ping -c 200 h3
```

```
mininet > h2 ping -c 200 h4
```

3. Captura de pacotes (Wireshark):

- Use o Wireshark para capturar os pacotes ICMP que fluem pelas interfaces do *switch* OpenFlow. Antes de enviar os pacotes no Mininet, primeiro você deve iniciar o Wireshark usando o seguinte comando no terminal mininet para abrir um terminal para o *switch* s1.

```
mininet > xterm s1
```

Para os hosts:

```
mininet > xterm s2
```

O comando mencionado anteriormente abrirá um novo terminal para o *switch* s1 e, em seguida, forneça o seguinte comando para iniciar o Wireshark:

wireshark &

Agora você verá a GUI do Wireshark, onde deverá definir o filtro para pacotes ICMP e selecionar a interface do *switch* s1 para capturar os pacotes ICMP. Existe uma opção na interface gráfica do Wireshark para iniciar a captura de pacotes durante a troca de mensagens entre *hosts* (h1 a h3 ou h2 a h4). Após a conclusão da troca de mensagens entre *hosts* é necessário interromper a captura dos pacotes. Em seguida, você pode salvar arquivos de captura de pacotes ICMP no formato PCAP (*Packet Capture*) para análise posterior desses arquivos.

4. Fazendo cópia dos arquivos gerados (Máquina virtual to host)

```
scp <caminho_da_vm>/<nome_do_arquivo> ra<seuRA>@ssh.lab.ic.unicamp.br:/<caminho_do_host>
```

- <caminho_da_vm>: diretório na VM onde está o arquivo a ser enviado
- <nome_do_arquivo>: nome do arquivo que você deseja copiar
- <seuRA>: seu número de RA na Unicamp
- <caminho_do_host>: diretório de destino no host onde o arquivo será salvo

Exemplo:

```
scp /home/aluno/relatorio.txt ra123456@ssh.lab.ic.unicamp.br:/home/ra123456/
```

Em caso de erro na cópia de arquivos

No terminal da máquina virtual, utilize o seguinte comando para descobrir o IP da máquina:

hostname -I

Substitua `ssh.lab.ic.unicamp.br` pelo endereço IP obtido no comando anterior, ao executar o `scp`.


5. Analisando dados (Python):

Desenvolva um script Python, utilizando a biblioteca Scapy, para analisar os dados do pacote capturado (arquivos PCAP). Extraia informações como:

- Endereços IP de origem e de destino.
- Calcular o *throughput* (taxa de transferência) médio.
- Intervalo médio entre pacotes (tempo entre chegadas de pacotes).
- Contagem de pacotes (total de pacotes).
- Além disso, o script deve gerar gráficos ilustrativos (usando matplotlib) mostrando claramente as métricas obtidas.

Como analisar pacotes de rede em arquivos .pcap com Python

Para analisar pacotes capturados com ferramentas como o Wireshark, podemos utilizar a biblioteca Scapy em Python. Isso permite contar, filtrar e manipular diferentes tipos de pacotes de rede.

 Instalação de bibliotecas necessárias

Antes de começar, certifique-se de instalar as bibliotecas necessárias:

```
pip install scapy nest_asyncio pandas numpy
```

Essas bibliotecas permitem a leitura de pacotes e a análise com suporte a estruturas de dados e processamento assíncrono.

 Exemplo: Contar pacotes TCP em um arquivo .pcap

6. Como analisar pacotes de rede em arquivos .pcap com Python

Exemplo: Contar pacotes TCP em um arquivo .pcap

```
from scapy.all import *
import nest_asyncio
nest_asyncio.apply()

def contar_pacotes_tcp(arquivo_pcap):
    pacotes = rdpcap(arquivo_pcap)
    pacotes_tcp = [pkt for pkt in pacotes if TCP in pkt]
    return len(pacotes_tcp)

if __name__ == "__main__":
    arquivo_pcap = "seu_arquivo.pcap"
    total_tcp = contar_pacotes_tcp(arquivo_pcap)
    print("Número total de pacotes TCP:", total_tcp)
```

Resultados (relatório):

Escreva um relatório de projeto bem estruturado contendo:

- Código Python completo comentado e explicado.
- Capturas de tela comentadas mostrando os resultados obtidos (endereços IP, throughput, intervalos entre pacotes, taxas de perda e retransmissões).
- Capturas de tela detalhadas do Wireshark mostrando claramente o tráfego capturado com filtros utilizados.

▪ **Envie também arquivos de captura do Wireshark (PCAP) contendo tráfego de rede.**

Data de entrega: 7 de maio