

Trabalho 3

Monitoramento Ativo e Passivo

Conrado Boeira¹

¹Laboratório de Redes de Computadores
Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)

1. Introdução

Operadores de rede costumam usar ferramentas de monitoramento de suas redes para poder identificar defeitos, aspectos de melhora e ter uma ideia geral do funcionamento de uma rede. Desta maneira, neste trabalho, foram desenvolvidos dois monitoradores diferentes, um passivo e um ativo. Ambos foram desenvolvidos usando a linguagem Python 3.8.5.

Na Secção 2, é apresentada a topologia de simulação criada para os testes. Na Secção 3 e 4 é explicado o funcionamento dos dois monitoradores. Finalmente, na Secção 5 são apresentados alguns testes executados e na Secção 6, as considerações finais.

2. Topologia

Para o desenvolvimento e teste dos programas desenvolvidos, foi criada uma topologia de redes de computadores dentro do simulador Core Network Emulator. Uma rede com 12 hosts, 4 subredes e 2 roteadores foi criada como pode ser visto na Figura1. Todos os endereços de hosts e interfaces de roteadores foram definidos estaticamente.

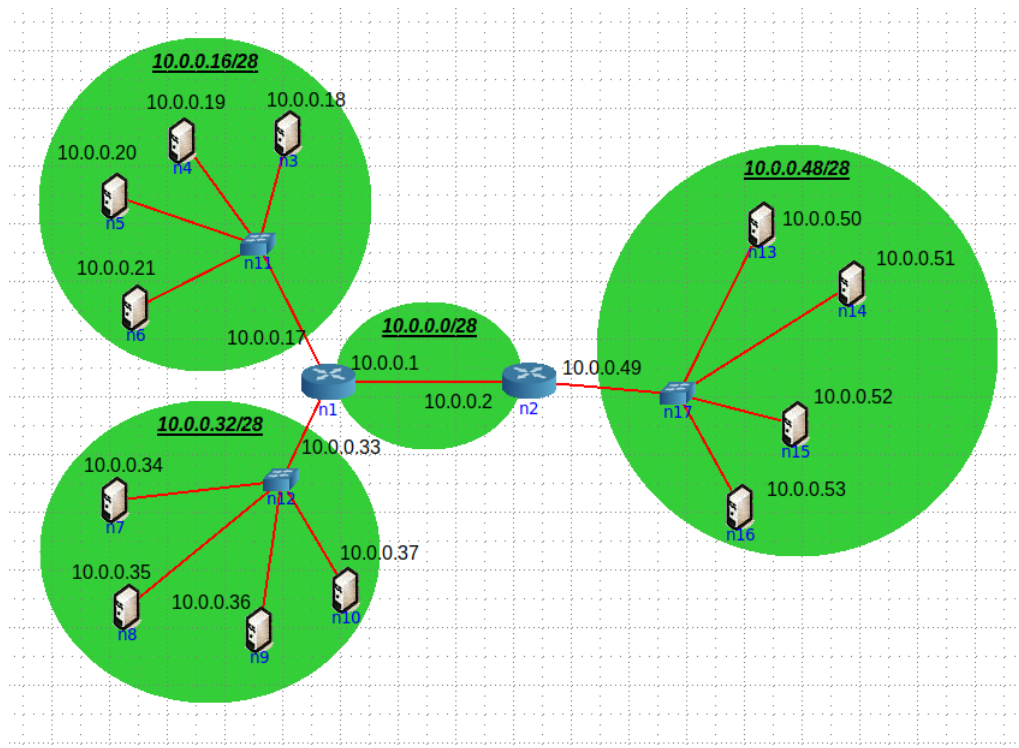


Figura 1. Topologia criada no Core Network Emulator

3. Monitorador Passivo

O monitorador passivo foi desenvolvido de maneira a coletar as informações dos cabeçalhos de todos pacotes Ethernet que passam pelo host que o executa. Para isso, ele primeiro retira o cabeçalho Ethernet do pacote para coletar informações sobre o MAC destino e origem e o do tipo de pacote. A partir daí, se o pacote for do tipo ARP, são exibidas informações do tipo de ARP, Reply ou Request, e os endereços IP dos computadores que trocam mensagens. Se o pacote for do tipo IP, são exibidas também informações dos endereços e o tempo de vida pacote. Neste caso, o pacote é novamente examinado para retirar os cabeçalhos dos protocolos que usam IP. Quando o pacote usa o protocolo ICMP, informações sobre o tipo de ICMP, o número de sequência e a identificação do pacote são exibidas. São apresentadas, também, as portas de origem e destino, quando o protocolo é UDP ou TCP.

Durante a execução, o monitorador guarda dados de todos pacotes para que, quando terminada sua execução, ele possa exibir diversas estatísticas. Ele apresenta valores para o total de pacotes capturados, o maior e o menor pacote e a porcentagem dos pacotes que pertence a cada protocolo. Além disso, são exibidos os endereços IPs e as portas que mais enviaram e receberam pacotes, além dos pares de hosts que mais trocaram mensagens entre si.

Para executar o programa basta usar o comando:

```
python3 monitor.py
```

4. Monitorador Ativo

O monitorador ativo realiza um scan de uma subrede informada com o intuito de descobrir quais endereços e portas estão sendo usados. Ele se divide em duas partes, o scan horizontal e vertical.

No scan horizontal, o monitorador tenta descobrir quais endereços IP estão sendo utilizados na rede alvo. Existem dois casos para isso, o que o host que executa o monitorador está na subrede alvo, e o caso em que ele não está. No primeiro caso, são enviados pacotes de ARP Request para todos os endereços da rede. Após o envio, é esperado o pacote de resposta por um tempo limite definido (0.5 segundos na implementação proposta) e caso não haja uma resposta, o endereço é considerado inativo. No caso em que a rede alvo é diferente da local, são usados ICMP Echo Requests para identificar se o endereço está sendo utilizado. Após o envio do ping, o host espera por uma resposta de Echo Reply ou de Host Unreachable para definir se o endereço é utilizado ou não.

No scan vertical, todas as portas dentro do intervalo desejado dos host encontrados como ativos são testadas com pacotes TCP e UDP. Primeiramente é feita a tentativa de conexão com pacotes de TCP SYN. Quando a porta não está ativa, a conexão não pode ser estabelecida e pacotes de TCP RST são recebidos. Depois deste teste, é enviado um pacote UDP para essa porta. O scan então espera por um tempo limite definido (0.5 segundos na implementação). Caso, neste meio tempo, o monitorador receba um pacote de ICMP Port Unreachable, a porta é considerada inativa. Caso o intervalo de tempo seja completo sem o recebimento deste pacote, a porta é considerada ativa.

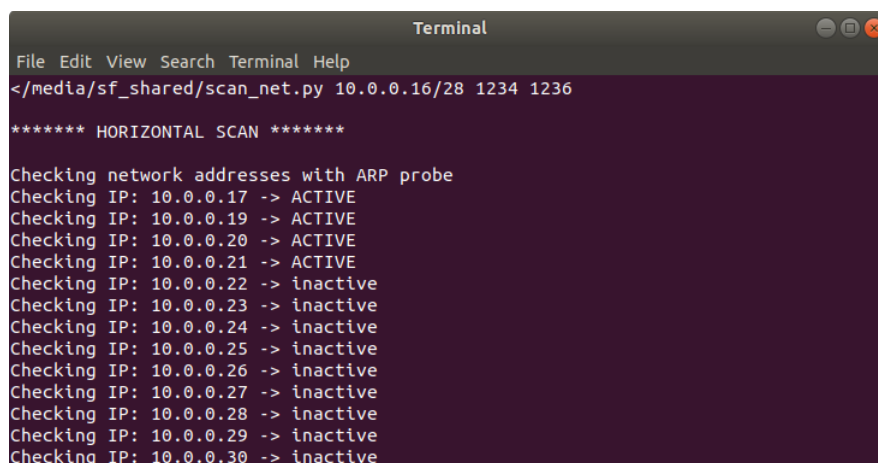
Para executar este monitorador, basta executar o comando:

```
python3 scan_net.py <subnet> <min_port> <max_port>
```

Onde subnet é o endereço da rede alvo com formato X.X.X.X/X e min_port e max_port definem os limites do intervalo de portas a serem testadas.

5. Testes

Para os testes, foi executado o programa de monitoramento passivo a partir do host n3 (IP 10.0.0.18). Na Figura 2 são exibidos os resultados da o scan horizontal na rede local. É possível de ver então o uso de ARP Requests para o estabelecimento dos host ativos e inativos.

A terminal window titled "Terminal" with a menu bar (File, Edit, View, Search, Terminal, Help). The command executed is "</media/sf_shared/scan_net.py 10.0.0.16/28 1234 1236". The output shows a "HORIZONTAL SCAN" with a list of IP addresses from 10.0.0.17 to 10.0.0.30. The first four IPs (10.0.0.17, 10.0.0.19, 10.0.0.20, 10.0.0.21) are marked as "ACTIVE", while the remaining 26 IPs are marked as "inactive".

```
Terminal
File Edit View Search Terminal Help
</media/sf_shared/scan_net.py 10.0.0.16/28 1234 1236

***** HORIZONTAL SCAN *****

Checking network addresses with ARP probe
Checking IP: 10.0.0.17 -> ACTIVE
Checking IP: 10.0.0.19 -> ACTIVE
Checking IP: 10.0.0.20 -> ACTIVE
Checking IP: 10.0.0.21 -> ACTIVE
Checking IP: 10.0.0.22 -> inactive
Checking IP: 10.0.0.23 -> inactive
Checking IP: 10.0.0.24 -> inactive
Checking IP: 10.0.0.25 -> inactive
Checking IP: 10.0.0.26 -> inactive
Checking IP: 10.0.0.27 -> inactive
Checking IP: 10.0.0.28 -> inactive
Checking IP: 10.0.0.29 -> inactive
Checking IP: 10.0.0.30 -> inactive
```

Figura 2. Execução de um scan horizontal na rede local

Para um teste mais completo, foram criados dois servidores, um que responde a TCP e outro a UDP. O servidor TCP está situado no host n7 (IP 10.0.0.34) usando a porta 1003. Já o UDP está no host n8 (IP 10.0.0.35) e usa a porta 1001. O monitorador passivo é então ativado no roteador n1. Finalmente, o host n3 executa o scan da rede usando o seguinte comando:

```
python3 scan_net.py 10.0.0.32/28 1000 1004
```

É possível ver na Figura 3, os hosts descobertos durante o scan horizontal. Já na Figura 4, é exibido o processo de scan vertical, sendo encontradas as duas portas abertas.

```
Terminal
File Edit View Search Terminal Help
</media/sf_shared/scan_net.py 10.0.0.32/28 1000 1004

***** HORIZONTAL SCAN *****

Checking network addresses with ICMP Request probe
Checking IP: 10.0.0.33 -> ACTIVE RTT: 0.024080 msec
Checking IP: 10.0.0.34 -> ACTIVE RTT: 0.014305 msec
Checking IP: 10.0.0.35 -> ACTIVE RTT: 0.012398 msec
Checking IP: 10.0.0.36 -> ACTIVE RTT: 0.011921 msec
Checking IP: 10.0.0.37 -> ACTIVE RTT: 0.011921 msec
Checking IP: 10.0.0.38 -> inactive RTT: 3050.212860 msec
Checking IP: 10.0.0.39 -> inactive RTT: 3071.516752 msec
Checking IP: 10.0.0.40 -> inactive RTT: 3071.755171 msec
Checking IP: 10.0.0.41 -> inactive RTT: 3071.630716 msec
Checking IP: 10.0.0.42 -> inactive RTT: 3071.432829 msec
Checking IP: 10.0.0.43 -> inactive RTT: 3071.542025 msec
Checking IP: 10.0.0.44 -> inactive RTT: 3071.676016 msec
Checking IP: 10.0.0.45 -> inactive RTT: 3071.723461 msec
Checking IP: 10.0.0.46 -> inactive RTT: 3071.468115 msec
```

Figura 3. Scan horizontal de uma rede não local

```
Terminal
File Edit View Search Terminal Help

TCP: -> inactive
UDP: -> inactive
Checking 10.0.0.34:1000
TCP: -> inactive
UDP: -> inactive
Checking 10.0.0.34:1001
TCP: -> inactive
UDP: -> inactive
Checking 10.0.0.34:1002
TCP: -> inactive
UDP: -> inactive
Checking 10.0.0.34:1003
TCP: -> ACTIVE
UDP: -> inactive
Checking 10.0.0.34:1004
TCP: -> inactive
UDP: -> inactive
Checking 10.0.0.35:1000
TCP: -> inactive
UDP: -> inactive
Checking 10.0.0.35:1001
TCP: -> inactive
UDP: -> ACTIVE
Checking 10.0.0.35:1002
```

Figura 4. Scan vertical de portas nos hosts ativos

O funcionamento do monitorador passivo é mostrado na Figura 5, onde é possível ver o momento que o scan envia um pacote UDP para uma porta fechada e recebe como resposta um pacote de Host Unreachable. Na Figura 6 são vistas as estatísticas coletadas durante o teste feito.

```
Terminal
File Edit View Search Terminal Help
Source Port: 50259
Destination Port: 1004

***** NEW PACKET *****
Ethernet Header Info:
Source MAC Address: 00:00:00:aa:00:09
Destination MAC Address: 00:00:00:aa:00:08
Packet Type: IP
-----
IP Header Info:
Source IP Address: 10.0.0.18
Destination IP Address: 10.0.0.37
Time To Live: 63
Encapsulated Protocol: UDP
-----
UDP Header Info:
Source Port: 50259
Destination Port: 1004

***** NEW PACKET *****
Ethernet Header Info:
Source MAC Address: 00:00:00:aa:00:08
Destination MAC Address: 00:00:00:aa:00:09
Packet Type: IP
-----
IP Header Info:
Source IP Address: 10.0.0.37
Destination IP Address: 10.0.0.18
Time To Live: 64
Encapsulated Protocol: ICMP
-----
ICMP Header Info:
ICMP Type: Port Unreachable

***** NEW PACKET *****
Ethernet Header Info:
Source MAC Address: 00:00:00:aa:00:04
Destination MAC Address: 00:00:00:aa:00:00
Packet Type: IP
```

Figura 5. Monitorador passivo capturando a troca de pacotes

```
Terminal
File Edit View Search Terminal Help
^C
***** STATS *****

Total packets captured: 218
Largest packet captured: 1042 bytes
Smallest packet captured: 42 bytes

Percentage of packets per each protocol
ARP : 22.477%
IP : 73.394%
ICMP : 36.239%
TCP : 15.596%
UDP : 21.560%
Undefined : 4.128%

IPs that send the most amount of packets
10.0.0.18 -> 85 packets
10.0.0.33 -> 46 packets
10.0.0.34 -> 24 packets
10.0.0.35 -> 14 packets
10.0.0.36 -> 14 packets

IPs that received the most amount of packets
10.0.0.18 -> 81 packets
10.0.0.34 -> 28 packets
10.0.0.33 -> 19 packets
10.0.0.35 -> 14 packets
10.0.0.36 -> 14 packets

Pair of IPs that exchanged the most amount of packets
10.0.0.18 x 10.0.0.34 -> 26 packets
10.0.0.34 x 10.0.0.18 -> 22 packets
10.0.0.18 x 10.0.0.35 -> 12 packets
10.0.0.35 x 10.0.0.18 -> 12 packets
10.0.0.18 x 10.0.0.36 -> 12 packets

Port that sent the most amount of packets: 50259
Port that received the most amount of packets: 1003
root@n1:/tmp/pycore.32887/n1.conf#
```

Figura 6. Estatísticas coletadas durante o teste

6. Conclusão

Neste trabalho foram explorados dois diferentes tipos de monitoradores de rede. Um que trabalha de forma passiva, analisando pacotes e recebidos e coletando estatísticas e outro, que ativamente escaneia a rede procurando endereços utilizados e portas abertas.