PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL

ESCOLA POLITÉCNICA

LABORATÓRIO DE REDES DE COMPUTADORES

TRABALHO FINAL – MONITORAÇÃO DE TRÁFEGO

Lara Alves Kunrath Padilha, Leonardo Garcia Forquim Bertinatto

1. Introdução

Este relatório apresenta o desenvolvimento de um monitor de tráfego de rede em tempo real, implementado utilizando **sockets RAW** em linguagem C, conforme especificado no trabalho final da disciplina Fundamentos de Redes de Computadores.

O objetivo principal deste trabalho é capturar pacotes diretamente da interface de rede antes que o sistema operacional os processe, permitindo identificar protocolos das camadas **Internet**, **Transporte** e **Aplicação**, além de registrar informações relevantes em arquivos CSV.

A ferramenta também exibe uma interface modo texto com contadores que são atualizados continuamente, permitindo observar o comportamento do tráfego em tempo real.

2. Desenvolvimento

A ferramenta foi desenvolvida em linguagem C utilizando chamadas de sistema do Linux, em especial:

- socket(AF_PACKET, SOCK_RAW, htons(ETH_P_ALL))
- setsockopt(... SO_BINDTODEVICE ...)
- recvfrom()
- select()

O uso de **AF_PACKET + SOCK_RAW** permite ao programa receber pacotes brutos diretamente da camada de enlace. Esses pacotes incluem cabeçalhos Ethernet, IPv4, IPv6, TCP, UDP e ICMP.

O programa recebe como argumento o nome da interface de rede que será monitorada. Exemplo:

sudo ./netmon enp2s0

Após iniciado, o fluxo interno faz as seguintes etapas:

1. Abre o raw socket e prende-o a uma interface.
2. Entra em um loop contínuo utilizando select() para aguardar pacotes.
3. Ao receber um pacote, identifica o offset do cabeçalho IP.
4. Decodifica os seguintes protocolos:
    - **Camada Internet:** IPv4, IPv6, ICMP
    - **Camada Transporte:** TCP, UDP
    - **Camada Aplicação:** HTTP, DNS, DHCP, NTP (por heurística de portas)
5. Registra as informações em três arquivos CSV:
    - internet.csv
    - transporte.csv
    - aplicacao.csv
6. Atualiza a interface modo texto com os contadores de pacotes.

Essa implementação atende ao requisito do trabalho: **não utiliza nenhuma biblioteca externa** como libpcap, Scapy ou similares.

Todo o parsing é feito manualmente a partir dos bytes capturados.

3. Arquivos de Log

Os arquivos gerados possuem os seguintes formatos:

internet.csv

Contém informações das camadas IPv4, IPv6 e ICMP:

- Timestamp
- Protocolo (IPv4/IPv6/ICMP)
- IP origem
- IP destino
- Protocolo carregado
- Tamanho total

transporte.csv

Contém informações de TCP e UDP:

- Timestamp
- Protocolo (TCP/UDP)
- IP origem
- Porta origem
- IP destino
- Porta destino
- Tamanho total

aplicacao.csv

Contém protocolos de aplicação detectados por portas:

- HTTP
- DNS
- DHCP
- NTP

Esses arquivos são atualizados em tempo real enquanto o monitor está em execução.

4. Testes Realizados

Para validar o comportamento da ferramenta, foram executados comandos que geram tráfego associado a diferentes protocolos:

- ping 8.8.8.8 → ICMP
- curl http://example.com → TCP + HTTP
- dig google.com → UDP + DNS
- sudo ntpdate pool.ntp.org → UDP + NTP
- sudo dhclient → UDP + DHCP

5. Conclusão

O desenvolvimento deste projeto permitiu uma compreensão prática sobre o funcionamento interno dos **raw sockets**, bem como dos formatos dos diferentes cabeçalhos de rede.