

Packet Sniffing Using raw Socket

AIM:

To study packet sniffing concept and implement it using raw sockets.

DESCRIPTION:

INTRODUCTION TO PACKET SNIFFING

- Packet sniffing is the act of capturing packets of data flowing across a computer network. The software or device used to do this is called a packet sniffer.
- In network management, packet sniffing plays a very crucial role. Network managers and technicians use packet sniffers to diagnose underlying problems in their networks. ● Packet sniffer is essentially a tool that aids in monitoring network traffic and troubleshooting a network.
- It works by capturing and analyzing packets of data that flow through a particular network.

How Do Packet Sniffers Work?

- To understand how a packet sniffer works, first understand that data travels through a network in the form of packets.
- In packet-switched networks, the data to be transmitted is broken down into several packets. These packets are reassembled once all the data packets reach their intended destination.
- When a packet sniffer is installed in the network, the sniffer intercepts the network traffic and captures the raw data packets.
- Subsequently, the captured data packet is analyzed by the packet sniffing software and presented to the network manager/technician in a user-friendly format.
- By user-friendly, it means the Network Administrator should be able to make sense of it.

Uses And Implementations Of Packet Sniffers

- Monitoring network usage

Packet sniffers are great at monitoring the network usage at any given time, helping Network Managers identify whether a particular network is normal or congested. Also, making it possible to identify bottlenecks within the network and identify and improve the performance with infrastructure upgrades.

- Identifying problems

Packet sniffers can identify network-related issues. This is possible because a packet sniffer can analyze the conversation between two or more nodes in a network. So, in the event of a network error, the information captured by the

[illegible]


```

import struct
import socket
import binascii

s = socket.socket(socket.AF_PACKET, socket.SOCK_RAW, socket.htons(0x0800))
#s.bind(("127.0.0.1", 0))
packet = s.recvfrom(65565)
print(packet)
ethernet_header = packet[0][0:14]
eth_header = struct.unpack("!6s6s2s", ethernet_header)
print("ETHERNET HEADER")
print("*****")
print("Destination Address")
print(binascii.hexlify(eth_header[0]))
print("Source Address")
print(binascii.hexlify(eth_header[1]))
print("Type")
print(binascii.hexlify(eth_header[2]))
print("IP HEADER")
print("*****")
ipheader = packet[0][14:34]

ip_header = struct.unpack("!12s4s4s", ipheader)
print("Destination Address")
print(socket.inet_ntoa(ip_header[1]))
print("Source Address")
print(socket.inet_ntoa(ip_header[2]))

```

OUTPUT:

```

(b'E\x00\x00\x00\x87@\x00\x80\x06\x00\x00\x7f\x00\x00\x01\x7f\x00\x00\x01\xd8_\x17a\x95\x08fs\x00\x00\x00\x00\x80\x02\xff\xff\x8b\xad\x00\x00\x02\x04\xff\xd7\x01\x03\x03\x08\x01\x01\x04\x02', ('127.0.0.1', 0))

```

ETHERNET HEADER

Destination Address

b'450000340087'

Source Address

b'400080060000'

Type

b'7f00'

IP HEADER

Destination Address

102.115.0.0

Source Address

0.0.128.2

```

(root@kali)-[/home/kali]
# python3 rawsocket.py
(b"\x08\x00'\xad%\x87RU\n\x00\x02\x02\x08\x00E\x00\x00\x99\x00\x03\x00\x00\x11b@\n\x00\x02\x03\n\x00\x02\x0f\x005\xe7H\x00\x85\x11\xdd\x98\xfd\x81\x80\x00\x01\x00\x02\x00\x00\x00\x00\x08location\x08services\x07mozilla\x03com\x00\x00\x01\x00\x01\xc0\x0c\x00\x05\x00\x01\x00\x00\x00\x0b\x002\x04prod\x0fclassify-client\x04prod\x0bwebservices\x06mozgcp\x03net\x00\xc0;\x00\x01\x00\x01\x00\x00\x00|\x00\x04#\xbeH\xd8", ('eth0', 2048, 0, 1, b'RU\n\x00\x02\x02'))
ETHERNET HEADER
*****
Destination Address
b'080027ad2587'
Source Address
b'52550a000202'
Type
b'0800'
IP HEADER
*****
Destination Address
10.0.2.3
Source Address
10.0.2.15

```

Result:

Thus a study on packet sniffing concept and implement it using raw sockets was done