Packet Sniffer Documentation

version

2025, Ajinkya Shetty

February 07, 2025

Contents

Welcome to Packet Sniffer Documentation!

Packet Sniffer Modules

pktsniffer module

Packet Sniffer

A simple packet sniffer for analyzing pcap files using Scapy. Supports filtering by host, port, protocol (TCP, UDP, ICMP), and network.

Usage:

python script.py -r file.pcap [options]

Author: [Your Name]

`pktsniffer.`**`packet_matches_filter`**(`pkt`, `args`)
Checks if a packet matches the user-specified filters.

Parameters:
- **pkt** (*scapy.packet.Packet*) – The packet to check.
- **args** (*argparse.Namespace*) – Parsed command-line arguments.

Returns: True if the packet matches the filter, False otherwise.

Return type: bool

`pktsniffer.`**`parse_arguments`**()
Parses command-line arguments using argparse.

Returns: Parsed command-line arguments.

Return type: argparse.Namespace

`pktsniffer.`**`parse_ethernet`**(`pkt`)
Parses Ethernet header details.

Parameters: **pkt** (*scapy.packet.Packet*) – The packet to parse.

Returns: (Packet size, Destination MAC, Source MAC, Ethertype)

Return type: tuple

`pktsniffer.`**`parse_ip`**(`pkt`)
Parses IP header details.

Parameters: **pkt** (*scapy.packet.Packet*) – The packet to parse.

Returns: (IP version, Header length, TOS, Total length, Identification, Flags,Fragment offset, TTL, Protocol, Checksum, Source IP, Destination IP)

Return type: tuple

`pktsniffer.`**`parse_transport`**(`pkt`)
Parses transport layer headers (TCP, UDP, ICMP).

Parameters: **pkt** (*scapy.packet.Packet*) – The packet to parse.

Returns: A formatted string describing the transport layer details.

Return type: str

`pktsniffer.`**`process_pcap`**(`file_path`, `args`)
Processes packets from a pcap file and applies user-specified filters.

Parameters:
- **file_path** (*str*) – Path to the .pcap file.
- **args** (*argparse.Namespace*) – Parsed command-line arguments.

Module Documentation

Packet Sniffer

A simple packet sniffer for analyzing pcap files using Scapy. Supports filtering by host, port, protocol (TCP, UDP, ICMP), and network.

Usage:

python script.py -r file.pcap [options]

Author: [Your Name]

`pktsniffer.`**`packet_matches_filter`**(`pkt`, `args`)
Checks if a packet matches the user-specified filters.

Parameters:
- **pkt** (*scapy.packet.Packet*) – The packet to check.
- **args** (*argparse.Namespace*) – Parsed command-line arguments.

Returns: True if the packet matches the filter, False otherwise.

Return type: bool

`pktsniffer.`**`parse_arguments`**()
Parses command-line arguments using argparse.

Returns: Parsed command-line arguments.

Return type: argparse.Namespace

`pktsniffer.`**`parse_ethernet`**(`pkt`)
Parses Ethernet header details.

Parameters: **pkt** (*scapy.packet.Packet*) – The packet to parse.

Returns: (Packet size, Destination MAC, Source MAC, Ethertype)

Return type: tuple

`pktsniffer.`**`parse_ip`**(`pkt`)
Parses IP header details.

Parameters: **pkt** (*scapy.packet.Packet*) – The packet to parse.

Returns: (IP version, Header length, TOS, Total length, Identification, Flags,Fragment offset, TTL, Protocol, Checksum, Source IP, Destination IP)

Return type: tuple

`pktsniffer.`**`parse_transport`**(`pkt`)
Parses transport layer headers (TCP, UDP, ICMP).

Parameters: **pkt** (*scapy.packet.Packet*) – The packet to parse.

Returns: A formatted string describing the transport layer details.

Return type: str

`pktsniffer.`**`process_pcap`**(`file_path`, `args`)
Processes packets from a pcap file and applies user-specified filters.

Parameters:
- **file_path** (*str*) – Path to the .pcap file.
- **args** (*argparse.Namespace*) – Parsed command-line arguments.

Index

M

P

Python Module Index

p