Unclassified

```
import pyshark
import sys
from ipaddress import ip_network, IPv4Address

def parse_pcap(pcap_file, attribute, ip_range=None, protocol=None):
    # Open the PCAP file for parsing
    capture = pyshark.FileCapture(pcap_file)

    # Iterate over each packet in the capture
    for packet in capture:
        # Check if the desired attribute is present in the packet
        if hasattr(packet, attribute):
            # Check if filtering by IP range is enabled
            if ip_range is not None:
                # Extract the source IP address
                source_ip = packet.ip.src

                # Check if the source IP is within the specified range
                if not is_ip_in_range(source_ip, ip_range):
                    continue

            # Check if filtering by protocol is enabled
            if protocol is not None:
                # Extract the protocol of the packet
                packet_protocol = packet.layers[1].layer_name

                # Check if the packet protocol matches the specified protocol
                if packet_protocol != protocol:
                    continue

            # Extract and print the attribute value
            value = getattr(packet, attribute)
            print(value)

    # Close the capture
    capture.close()

def is_ip_in_range(ip, ip_range):
    # Convert the IP address to IPv4Address object
    ip_address = IPv4Address(ip)

    # Iterate over the IP range networks
    for network in ip_range:
        if ip_address in network:
            return True

    return False

if __name__ == "__main__":
    # Check if the PCAP file path and attribute are provided as command-line
```

```
arguments
    if len(sys.argv) < 3:
        print("Usage: python pcap_parser.py <pcap_file> <attribute> [ip_range]
[protocol]")
        sys.exit(1)

    pcap_file = sys.argv[1]
    attribute = sys.argv[2]

    # Check if IP range is provided as command-line argument
    ip_range = None
    if len(sys.argv) >= 4:
        ip_range = [ip_network(sys.argv[3])]

    # Check if protocol is provided as command-line argument
    protocol = None
    if len(sys.argv) == 5:
        protocol = sys.argv[4]

    # Call the parse_pcap function
    parse_pcap(pcap_file, attribute, ip_range, protocol)
```

In this expanded version, we added the `protocol` parameter to the `parse_pcap` function. If a protocol is provided as a command-line argument, it checks if the packet protocol matches the specified protocol before extracting the attribute value. If the packet protocol doesn't match, the program moves on to the next packet without printing the attribute value.

You can run the program as before, providing the PCAP file path, the desired attribute, an optional IP address range, and an optional protocol as command-line arguments. For example:

```
python pcap_parser.py my_capture.pcap ip.src 192.168.0.0/24 tcp
```

This version of the program allows you to filter packets based on both IP address range and protocol type, in addition to extracting the desired attribute. Unclassified