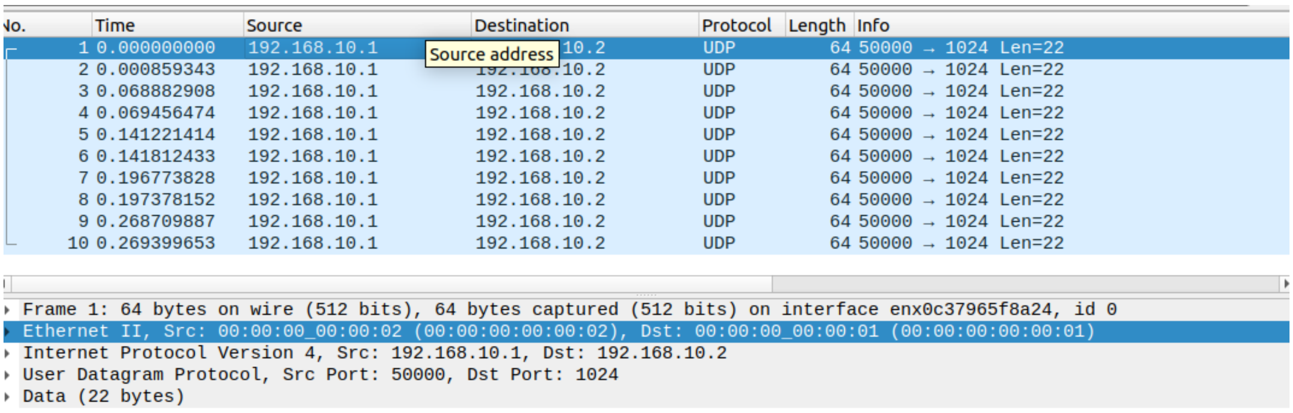
Exercise 3
Madeleine Rodriguez

Screenshot from wireshark:

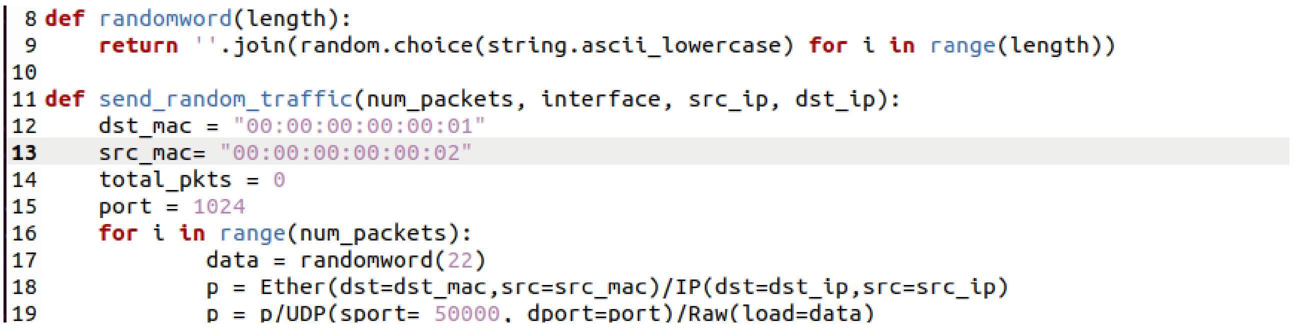


```
No.  Time         Source         Destination    Protocol Length Info
   1 0.000000000  192.168.10.1   192.168.10.2   UDP          64 50000 → 1024 Len=22
   2 0.000859343  192.168.10.1   192.168.10.2   UDP          64 50000 → 1024 Len=22
   3 0.068882908  192.168.10.1   192.168.10.2   UDP          64 50000 → 1024 Len=22
   4 0.069456474  192.168.10.1   192.168.10.2   UDP          64 50000 → 1024 Len=22
   5 0.141221414  192.168.10.1   192.168.10.2   UDP          64 50000 → 1024 Len=22
   6 0.141812433  192.168.10.1   192.168.10.2   UDP          64 50000 → 1024 Len=22
   7 0.196773828  192.168.10.1   192.168.10.2   UDP          64 50000 → 1024 Len=22
   8 0.197378152  192.168.10.1   192.168.10.2   UDP          64 50000 → 1024 Len=22
   9 0.268709887  192.168.10.1   192.168.10.2   UDP          64 50000 → 1024 Len=22
  10 0.269399653  192.168.10.1   192.168.10.2   UDP          64 50000 → 1024 Len=22

Frame 1: 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface enx0c37965f8a24, id 0
Ethernet II, Src: 00:00:00_00:00:02 (00:00:00:00:00:02), Dst: 00:00:00_00:00:01 (00:00:00:00:00:01)
Internet Protocol Version 4, Src: 192.168.10.1, Dst: 192.168.10.2
User Datagram Protocol, Src Port: 50000, Dst Port: 1024
Data (22 bytes)
```

Here I sent 5 packets from the lab machine to the raspberry pi and 5 were sent back to the lab machine as the valid ethernet addresses meant all of the packets were returned to the source.

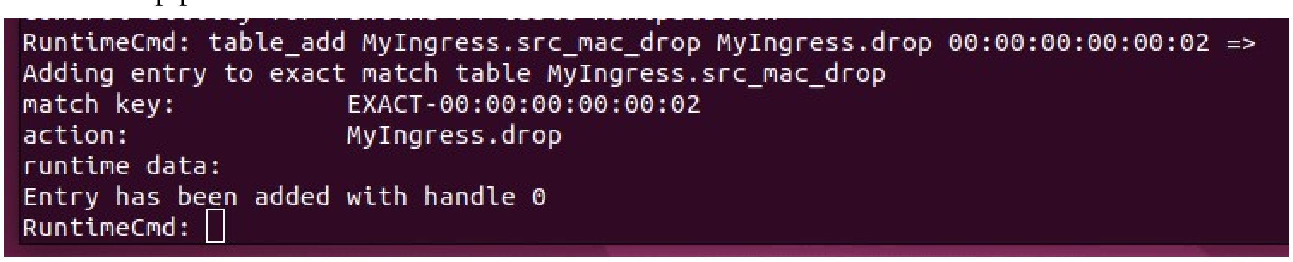
Table entry needed to drop traffic?

Adding 00:00:00:00:00:02 to be dropped to the table as this is the source mac address in send.py This source mac address can also be seen in wireshark as this was the source address of the packets sent to the raspberry pi and the destination address of the packets the raspberry pi sent back.



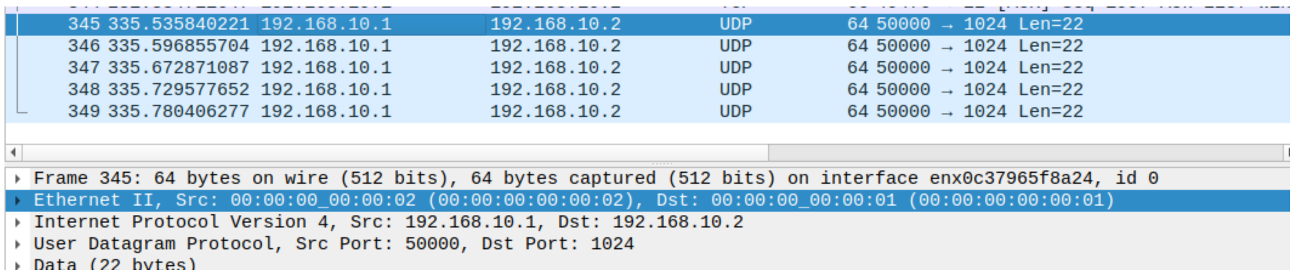
```python
def randomword(length):
    return ''.join(random.choice(string.ascii_lowercase) for i in range(length))

def send_random_traffic(num_packets, interface, src_ip, dst_ip):
    dst_mac = "00:00:00:00:00:01"
    src_mac= "00:00:00:00:00:02"
    total_pkts = 0
    port = 1024
    for i in range(num_packets):
            data = randomword(22)
            p = Ether(dst=dst_mac,src=src_mac)/IP(dst=dst_ip,src=src_ip)
            p = p/UDP(sport= 50000, dport=port)/Raw(load=data)
```

Code to drop packets from source mac address 00:00:00:00:00:02:



```
RuntimeCmd: table_add MyIngress.src_mac_drop MyIngress.drop 00:00:00:00:00:02 =>
Adding entry to exact match table MyIngress.src_mac_drop
match key:           EXACT-00:00:00:00:00:02
action:              MyIngress.drop
runtime data:
Entry has been added with handle 0
RuntimeCmd: 
```

Result of dropping packets from this mac address:



```
345 335.535840221 192.168.10.1      192.168.10.2      UDP      64 50000 → 1024 Len=22
346 335.596855704 192.168.10.1      192.168.10.2      UDP      64 50000 → 1024 Len=22
347 335.672871087 192.168.10.1      192.168.10.2      UDP      64 50000 → 1024 Len=22
348 335.729577652 192.168.10.1      192.168.10.2      UDP      64 50000 → 1024 Len=22
349 335.780406277 192.168.10.1      192.168.10.2      UDP      64 50000 → 1024 Len=22
```

```
Frame 345: 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface enx0c37965f8a24, id 0
Ethernet II, Src: 00:00:00_00:00:02 (00:00:00:00:00:02), Dst: 00:00:00_00:00:01 (00:00:00:00:00:01)
Internet Protocol Version 4, Src: 192.168.10.1, Dst: 192.168.10.2
User Datagram Protocol, Src Port: 50000, Dst Port: 1024
Data (22 bytes)
```

Five packets were received but none of them were sent back, as expected (the packets all had source mac address 00:00:00:00:00:02).