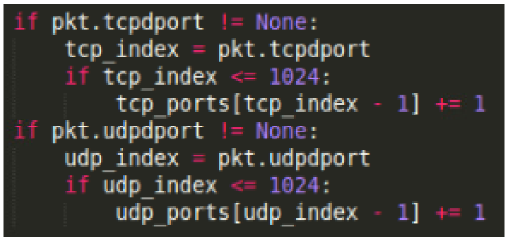
CS 373: Lab 2

Nathan Shepherd

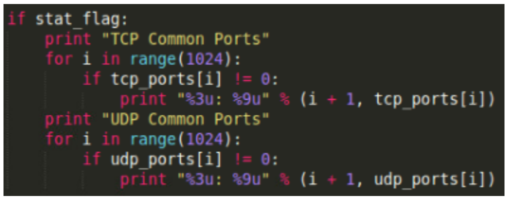
Procedure

1. In part 1, I wrote additional python to display the common UDP and TCP destination ports. It was easy enough to work around the CSV helper file, and count a total for both UDP and TCP. The only challenge I faced was making sure the None data type was accounted for. The following code is what I wrote in:



```
if pkt.tcpdport != None:
    tcp_index = pkt.tcpdport
    if tcp_index <= 1024:
        tcp_ports[tcp_index - 1] += 1
if pkt.udpdport != None:
    udp_index = pkt.udpdport
    if udp_index <= 1024:
        udp_ports[udp_index - 1] += 1
```

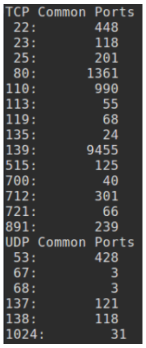
Figure 1: Counting code



```
if stat_flag:
    print "TCP Common Ports"
    for i in range(1024):
        if tcp_ports[i] != 0:
            print "%3u: %9u" % (i + 1, tcp_ports[i])
    print "UDP Common Ports"
    for i in range(1024):
        if udp_ports[i] != 0:
            print "%3u: %9u" % (i + 1, udp_ports[i])
```

Figure 2: Displaying Code

Below is the actual output of the R.csv file.



```
TCP Common Ports
 22:       448
 23:       118
 25:       201
 80:      1361
110:       990
113:        55
119:        68
135:        24
139:      9455
515:       125
700:        40
712:       301
721:        66
891:       239
UDP Common Ports
 53:       428
 67:         3
 68:         3
137:       121
138:       118
1024:        31
```

Figure 3: R file output

2. For the uses of each network I looked at the majority of the ports being used. For the O network, we see the following large uses:

- 22/TCP: 26,383…ssh
- 25/TCP: 211,205..smtp, mail
- 53/UDP: 21,563…domain
- 80/TCP: 156,397..http, WWW
- 445/TCP: 10,867..microsoft-ds, Microsoft Naked CIFS

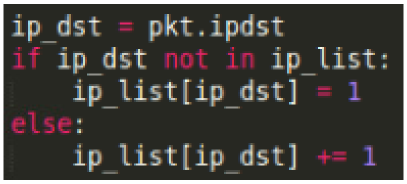
Looking these up, it seems that this network was used mainly for emails and web browsing. It most likely is a home network or a business.

For the R network, the following totals stand out:

- 53/UDP: 428….domain
- 139/TCP: 9455..netbios-ssn

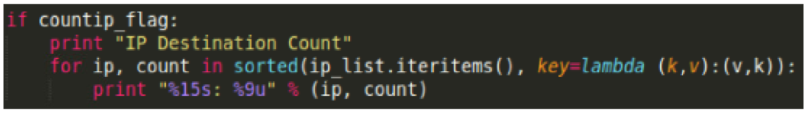
This network has much less traffic than the O network, but is somehow associated with file sharing, because that is the 139 port. I might assume that this is a small piece of a server or data center.

3. To count the ip addresses I added a dictionary, using the address as the key and the count as the value. This is the code I wrote:



```
ip_dst = pkt.ipdst
if ip_dst not in ip_list:
    ip_list[ip_dst] = 1
else:
    ip_list[ip_dst] += 1
```

Figure 4: Countip counting code



```
if countip_flag:
    print "IP Destination Count"
    for ip, count in sorted(ip_list.iteritems(), key=lambda (k,v):(v,k)):
        print "%15s: %9u" % (ip, count)
```

Figure 5: Countip display code

4. The code printed out the address counts and I was able to record the most requested addresses. For the O network, the following are the top three addresses:

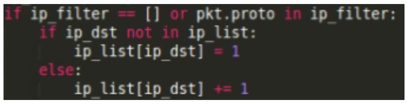
- 192.245.12.230 - total 60,866
- 192.245.12.242 - total 69,056
- 192.245.12.221 - total 118,662

These could be corporate lan addresses, which does sound like a business.

For the R network, the following are the top three addresses: - 10.5.63.231 - 6,974 - 10.5.63.230 - 16,073 - 234.142.142.142 - 42,981

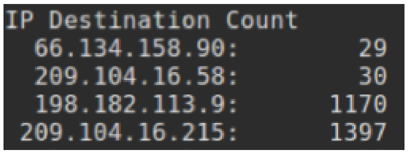
I'm not sure what the 10 subnet might mean, but 234.142.142.142 is used to copy files from servers to clients on a local network. This does assist my idea that it might be a small server, or business router.

5. The most common prefix in network R is 10.5.63, while the most common prefix in network O is 192.245.12.
6. I implemented the additional code, just making the counter ignore any filtered out protocols. Here is the code implemented:



```
if ip_filter == [] or pkt.proto in ip_filter:
    if ip_dst not in ip_list:
        ip_list[ip_dst] = 1
    else:
        ip_list[ip_dst] += 1
```

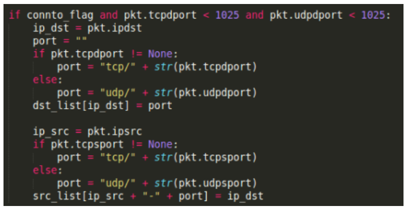
Figure 6: Countip with filtering code



```
IP Destination Count
  66.134.158.90:        29
  209.104.16.58:        30
  198.182.113.9:      1170
 209.104.16.215:      1397
```

Figure 7: Filtered on GRE

7. Running the countip function with filtering I found that the predominant prefix for the O network filtered on GRE is 209, filtered on IPSEC is 146, and filtered on OSPF is 207.182.35. When filtering the R network traffic on GRE, IPSEC, or OSPF, there turns out to be no traffic for any of those protocols.
8. With both the O network and the R network not having very many OSPF packets, I can guess that neither of them are routers. This is what I had earlier thought, so it does strengthen my guesses.
9. Although I did implement this section, either it didn't work, or I just wasn't able to understand the instructions. The code I added is below:



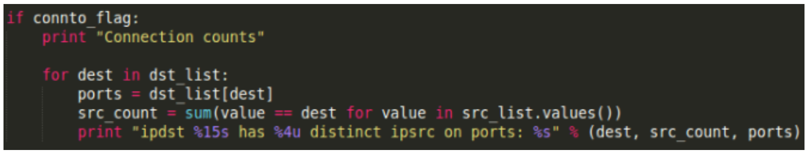
```
if connto_flag and pkt.tcpdport < 1025 and pkt.udpdport < 1025:
    ip_dst = pkt.ipdst
    port = ""
    if pkt.tcpdport != None:
        port = "tcp/" + str(pkt.tcpdport)
    else:
        port = "udp/" + str(pkt.udpdport)
    dst_list[ip_dst] = port

    ip_src = pkt.ipsrc
    if pkt.tcpsport != None:
        port = "tcp/" + str(pkt.tcpsport)
    else:
        port = "udp/" + str(pkt.udpsport)
    src_list[ip_src + "-" + port] = ip_dst
```

Figure 8: Connto counting code

With the code running, even if incorrectly, I managed to get the following output:

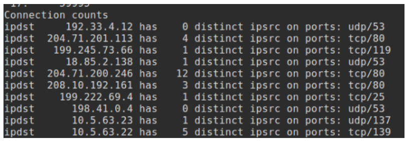
10. From the R network, the most connected to port is 10.5.63.6, which connects under UDP 53. This is a domain request, which makes sense if this is a web browsing computer. From the O network, the ip address 192.245.12.234 was the most connected to, using TCP on port 25. This makes sense if it is a mail server, or file transfer.



```python
if connto_flag:
    print "Connection counts"

    for dest in dst_list:
        ports = dst_list[dest]
        src_count = sum(value == dest for value in src_list.values())
        print "ipdst %15s has %4u distinct ipsrc on ports: %s" % (dest, src_count, ports)
```

Figure 9: Connto display code



```
Connection counts
ipdst     192.33.4.12 has    0 distinct ipsrc on ports: udp/53
ipdst  204.71.201.113 has    4 distinct ipsrc on ports: tcp/80
ipdst   199.245.73.66 has    1 distinct ipsrc on ports: tcp/119
ipdst     18.85.2.138 has    1 distinct ipsrc on ports: udp/53
ipdst  204.71.200.246 has   12 distinct ipsrc on ports: tcp/80
ipdst  208.10.192.161 has    3 distinct ipsrc on ports: tcp/80
ipdst    199.222.69.4 has    1 distinct ipsrc on ports: tcp/25
ipdst      198.41.0.4 has    0 distinct ipsrc on ports: udp/53
ipdst      10.5.63.23 has    1 distinct ipsrc on ports: udp/137
ipdst      10.5.63.22 has    5 distinct ipsrc on ports: tcp/139
```

Figure 10: Connto output