NCTU CN2018 Lab. 1 – Packet Manipulation via Scapy

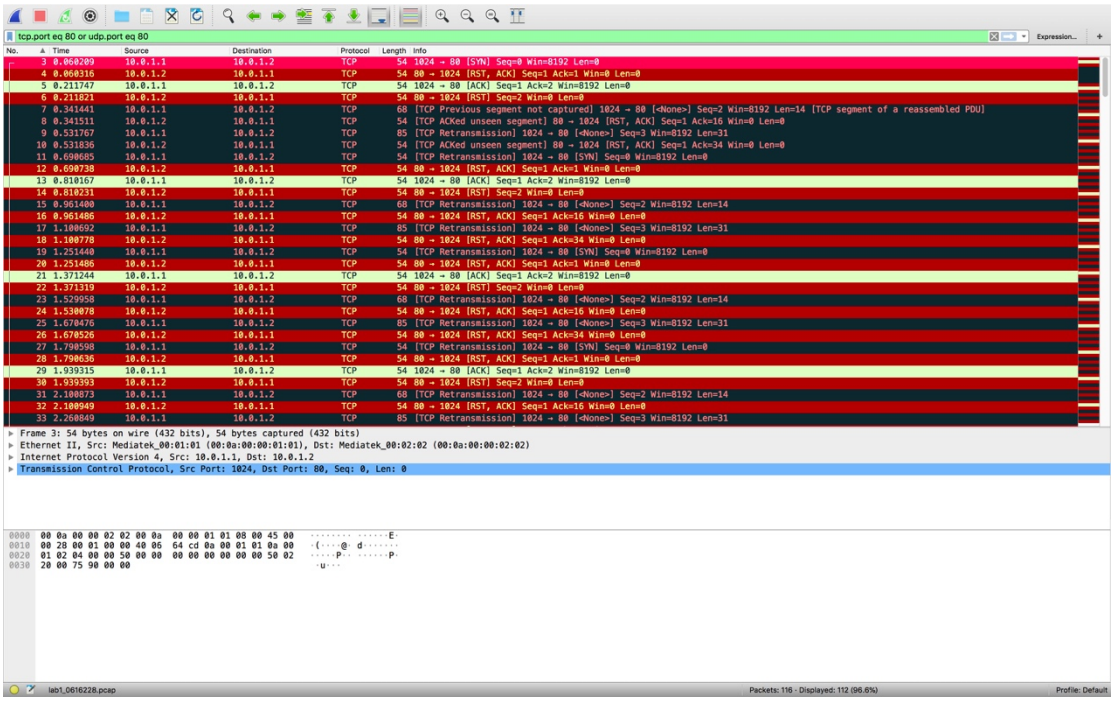
Student name: 莊于萱 Student ID: 0616228 Department: CS

Part A. Questions

1. What is your command to filter the packet with customized header on Wireshark?

   tcp.port eq 80 or udp.port eq 80

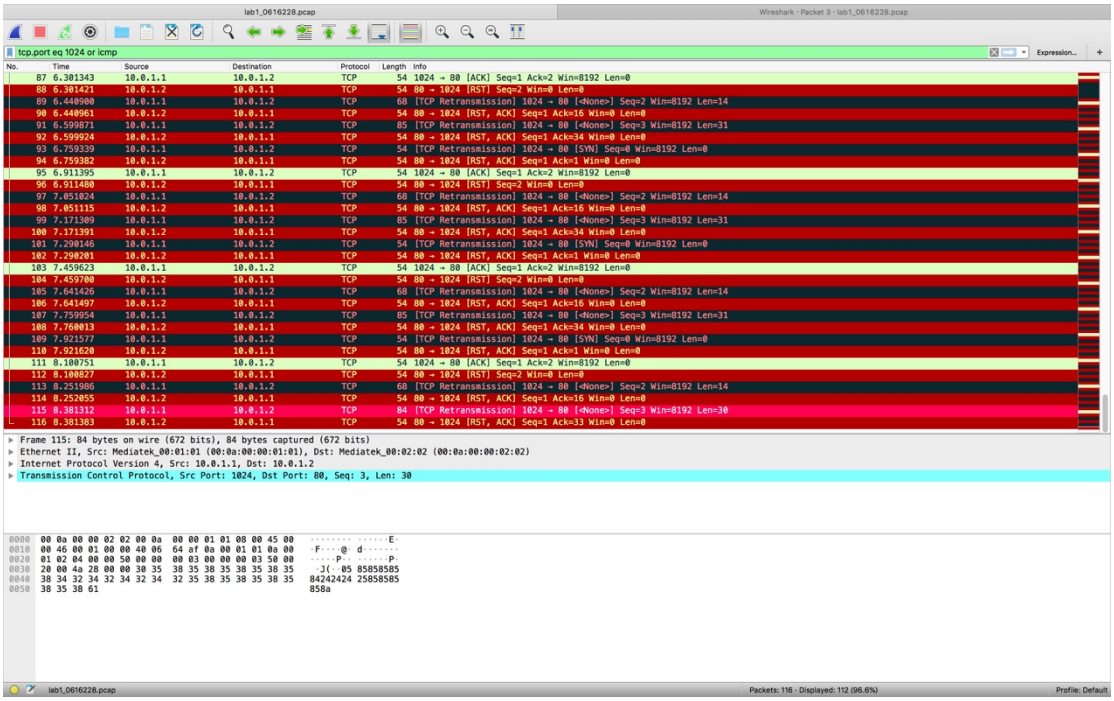
2. Show the screenshot of filtering the packet with customized header.



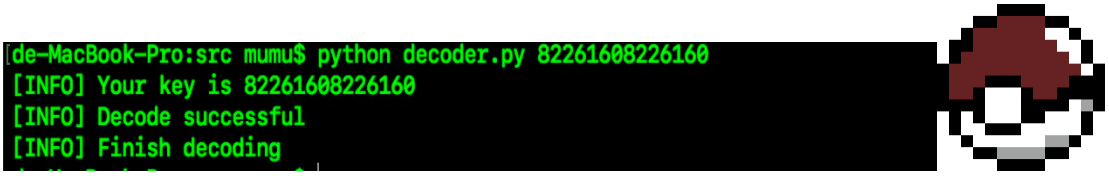
3. What is your command to filter the packet with "secret" payload on Wireshark?

   tcp.port eq 1024 or icmp

4. Show the screenshot of filtering the packet with "secret" payload .



5. Show the result after decoding the "secret" payload.



Part B. Description

Task 1 - **Environment setup**

Download required files from GitHub

Use $git clone https://github.com/yungshenglu/Packet_Manipulation

Set Dockerfile

```
# Download base image from yungshenglu/ubuntu-env:16.04 (Task 1.)
FROM yungshenglu/ubuntu-env:16.04

# Install software repository (Task 1.)
RUN apt-get update

# Install software repository (Task 1.)
RUN apt-get install -y tcpdump

# Install packages (Task 1.)
RUN pip install scapy

# Use argument to assign passwd to create image
RUN echo 'root:cn2018' | chpasswd
RUN sed -i 's/PermitRootLogin prohibit-password/PermitRootLogin yes/' /etc/ssh/sshd_config

# SSH login fix. Otherwise user is kicked off after login
RUN sed 's@session\s*required\s*pam_loginuid.so@session optional pam_loginuid.so@g' -i /etc/pam.d/sshd

# Set the envionment variables
ENV NOTVISIBLE "in users profile"
ENV LC_ALL C
RUN echo "export VISIBLE=now" >> /etc/profile

# Set the container listens on the specified ports at runtime (Task 1.)
EXPOSE 22

# Set the entrypoint
CMD ["/usr/sbin/sshd", "-D"]

# Clone the repository from GitHub (Task 1.)
RUN git clone https://github.com/yungshenglu/Packet_Manipulation.git
```

Open the Terminal and change the path to ./docker/ and build the environment as follows:

```
de-MacBook-Pro:docker mumu$ sudo chmod +x main.sh
Password:
de-MacBook-Pro:docker mumu$ ./main.sh build cn2018 9487
[INFO] Docker image: cn2018
[INFO] External port: 9487
Sending build context to Docker daemon  206.8kB
```

After finish 13 steps

```
Cloning into 'Packet_Manipulation'...
Removing intermediate container aed0d3284dd5
 ---> 206a15760879
Successfully built 206a15760879
Successfully tagged cn2018:latest
0.0.0.0:9487
```

Then Login to your Docker container using SSH

Use terminal to connect to the Docker

```
de-MacBook-Pro:docker mumu$ ssh -p 9487 root@0.0.0.0
The authenticity of host '[0.0.0.0]:9487 ([127.0.0.1]:9487)' can't be established.
ECDSA key fingerprint is SHA256:ieHrh9dG+pET1XFT3TmhDbm/NfQgT580XxH0tzB1180.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[0.0.0.0]:9487' (ECDSA) to the list of known hosts.
root@0.0.0.0's password:
Welcome to Ubuntu 16.04.5 LTS (GNU/Linux 4.9.93-linuxkit-aufs x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

Create the namespace in ./src/scripts/main.sh for h2

```
18  function addns {
19      echo "[INFO] Create h1 and h2 network namespaces"
20      ip netns add h1
21      # Create h2 network namespaces (Task 1.)
22      ip netns add h2
23  }
24
25  function delns {
26      echo "[INFO] Delete h1 and h2 network namespaces"
27      ip netns del h1
28      # Delete h2 network namespaces (Task 1.)
29      ip netns del h2
30  }
31
32  function lookup {
33      echo "[INFO] Bring up the lookup interface in h1 and h2"
34      ip netns exec h1 ip link set lo up
35      # Bring up the lookup interface in h2 (Task 1.)
36      ip netns exec h2 ip link set lo up
37  }
38
39  function addlink {
40      echo "[INFO] Build the link: h1-eth0 <-> h2-eth0"
41      ip link add h1-eth0 type veth peer name h2-eth0
42      ip link set h1-eth0 netns h1
43      # Set the interface of h2 to h2-eth0 (Task 1.)
44      ip link set h2-eth0 netns h2
45  }
46
47  function dellink {
48      echo "[INFO] Delete the link: h1-eth0 <-> h2-eth0"
49      ip link delete h1-eth0
50      # Delete the interface of h2-eth0 (Task 1.)
51      ip link delete h2-eth0
52  }
```

```
function activate {
    echo "[INFO] Activate h1-eth0 and assign IP address"
    ip netns exec h1 ip link set dev h1-eth0 up
    ip netns exec h1 ip link set h1-eth0 address 00:0a:00:00:01:01
    ip netns exec h1 ip addr add 10.0.1.1/24 dev h1-eth0

    echo "[INFO] Activate h2-eth0 and assign IP address"
    # Activate h2-eth0 and assign IP address (Task 1.)
    ip netns exec h2 ip link set dev h2-eth0 up
    ip netns exec h2 ip link set h2-eth0 address 00:0a:00:00:02:02
    ip netns exec h2 ip addr add 10.0.1.2/24 dev h2-eth0
}

function disableIPv6 {
    echo "[INFO] Disable all IPv6 on h1-eth0 and h2-eth0"
    ip netns exec h1 sysctl net.ipv6.conf.h1-eth0.disable_ipv6=1
    # Disable all IPv6 on h2-eth0 (Task 1.)
    ip netns exec h2 sysctl net.ipv6.conf.h2-eth0.disable_ipv6=1
}

function route {
    echo "[INFO] Set the gateway to 10.0.1.254 in routing table"
    ip netns exec h1 ip route add default via 10.0.1.254
    # Set the gateway of h2 to 10.0.1.254 (Task 1.)
    ip netns exec h2 ip route add default via 10.0.1.254
}
```

Run main.sh to build the namespace

```
6b265807f888 root ~ Packet_Manipulation src scripts vim main.sh
6b265807f888 root ~ Packet_Manipulation src scripts chmod +x main.sh
6b265807f888 root ~ Packet_Manipulation src scripts ./main.sh net
[INFO] Create h1 and h2 network namespaces
[INFO] Bring up the lookup interface in h1 and h2
[INFO] Build the link: h1-eth0 <-> h2-eth0
[INFO] Activate h1-eth0 and assign IP address
[INFO] Activate h2-eth0 and assign IP address
[INFO] Disable all IPv6 on h1-eth0 and h2-eth0
net.ipv6.conf.h1-eth0.disable_ipv6 = 1
net.ipv6.conf.h2-eth0.disable_ipv6 = 1
[INFO] Set the gateway to 10.0.1.254 in routing table
```

Task2 - Define protocol via Scapy

Define your protocol: Define ID header format

Copy the following code to ./src/Protocol.py

```python
#!/usr/bin/env python

from scapy.all import *

'''
Define your own protocol
'''
class Protocol(Packet):
    # Set the name of protocol (Task 2.)
    name = 'Student'
    # Define the fields in protocol (Task 2.)
    fields_desc = [
        StrField('index', '0'),
        StrField('dept', 'cs', fmt = 'H', remain = 0),
    IntEnumField('gender', 2, {
            1: 'female',
        2: 'male'
    }),
    StrField('id', '000000', fmt = 'H', remain = 0),
    ]

'''
Add customized protocol into IP layer
'''
bind_layers(TCP, Protocol, frag = 0, proto = 99)
conf.stats_classic_protocols += [Protocol]
conf.stats_dot11_protocols += [Protocol]
```

Task3 - Send packets

Setup your own packet header in ./src/sender.py

```python
# Set source and destination IP address (Task 3.)
src_ip = '10.0.1.1'
dst_ip = '10.0.1.2'
# Set source and destination port (Task 3.)
src_port = 1024
dst_port = 80

'''
Main function
'''
def main():
    # Define IP header (Task 3.)
    ip = IP(src = src_ip, dst = dst_ip)

    # Define customized header (Task 3.)
    my_id = '0616228'
    my_dept = 'cs'
    my_gender = 1
    student = Protocol(id = my_id, dept = my_dept, gender = my_gender)
```

Add the codes below in ./src/sender.py

```
48          # TCP connection – ACK (Task 3.)
49          ack = tcp_syn_ack.seq + 1
50      tcp_ack = TCP(sport = src_port, dport = dst_port, flags =
51      'A', seq = 1, ack = ack)
52      packet = ip / tcp_ack
53      send(packet)
54      print '[INFO] Send ACK'
55
56          # Send packet with customized header (Task 3.)
57      ack = tcp_ack.seq + 1
58      tcp = TCP(sport = src_port, dport = dst_port, flags = '',
59      seq = 2, ack = ack)
60      packet = ip / tcp / student
61      send(packet)
62          print '[INFO] Send packet with customized header'
63
64
65          # Send packet with secret payload (Task 3.)
66          ack = tcp.seq + 1
67      tcp = TCP(sport = src_port, dport = dst_port, flags = '',
68      seq = 3, ack = ack)
69      payload = Raw(secret[i])
70      packet = ip / tcp / payload
71      send(packet)
72      print '[INFO] Send packet with secret payload'
```

Task4 - Sniff packets

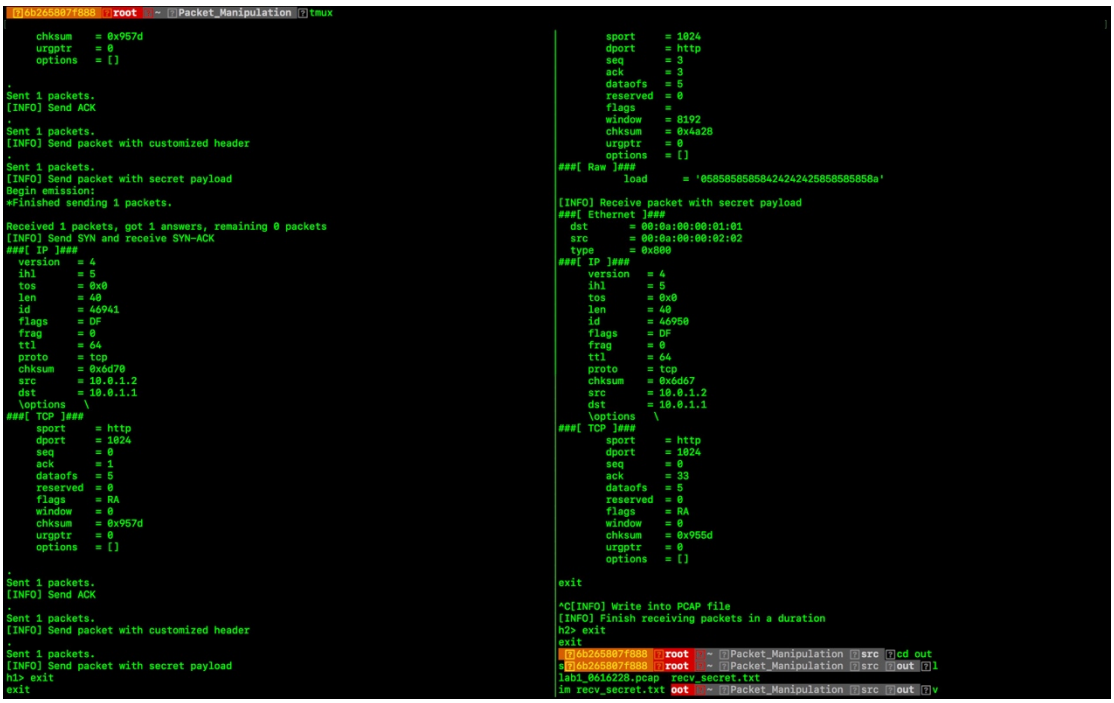
Receive and sniff packets: Add the codes below in ./src/receiver.py

```
7   # Set source IP address and destination interface (Task 4.)
8   dst_iface = 'h2-eth0'
9   src_ip = '10.0.1.1'
```

```
39  def main():
40      # Sniff packets on destination interface (Task 4.)
41      print '[INFO] Sniff on %s' % dst_iface
42      packets = sniff(iface = dst_iface, prn = lambda x:
43      packetHandler(x))
44
45      # Dump the sniffed packet into PCAP file (Task 4.)
46      print '[INFO] Write into PCAP file'
47      filename = './out/lab1_0' + id + '.pcap'
48      wrpcap(filename, packets)
```

Task5 - Run sender and receiver

Open tmux with horizontal two panes



Use tcpdump to show your PCAP file

```
de-MacBook-Pro:docker mumu$ tcpdump -qns 0 -X -r lab1_0616228.pcap
```

Task6 - Push your files to remote

Push my files to docker

```
de-MacBook-Pro:~ mumu$ docker commit cn2018_c yumumuu/cn2018_lab1
sha256:db3281c2250544b9f3a5fb1f3cb6ce1e72eee70ab33ebe3182d064a7d94b9aee
de-MacBook-Pro:~ mumu$ docker login
Authenticating with existing credentials...
Login Succeeded
de-MacBook-Pro:~ mumu$ docker push yumumuu/cn2018_lab1
The push refers to repository [docker.io/yumumuu/cn2018_lab1]
ff9d00914235: Preparing
5c2495954cef: Preparing
74581b08ef1c: Preparing
21d4286ee761: Preparing
213f64da6e1a: Preparing
7e24eb5f2cce: Preparing
83094da80dd1: Preparing
52a020b4006e: Preparing
68e1a36e78b4: Preparing
a56cf2693e71: Preparing
967459d06600: Preparing
83094da80dd1: Pushed
37935d3d3738: Mounted from yungshenglu/ubuntu-env
4f4d3617ed6b: Mounted from yungshenglu/ubuntu-env
8bf4df61381d: Mounted from yungshenglu/ubuntu-env
8a3273968425: Mounted from yungshenglu/ubuntu-env
990d06d9bcde: Mounted from yungshenglu/ubuntu-env
b7a36dab0729: Mounted from yungshenglu/ubuntu-env
01c983c58946: Mounted from yungshenglu/ubuntu-env
9dc1de5cb96d: Mounted from yungshenglu/ubuntu-env
e98a4c9eacd0: Mounted from yungshenglu/ubuntu-env
b280dcba3c7c: Mounted from yungshenglu/ubuntu-env
b36d26ea1891: Mounted from yungshenglu/ubuntu-env
f1d658477702: Mounted from yungshenglu/ubuntu-env
75b60323cb5f: Mounted from yungshenglu/ubuntu-env
cb8256e676a3: Mounted from yungshenglu/ubuntu-env
3db5b37d6167: Mounted from yungshenglu/ubuntu-env
08670dc9fd71: Mounted from yungshenglu/ubuntu-env
4d495c66c320: Mounted from yungshenglu/ubuntu-env
75b79e19929c: Mounted from yungshenglu/ubuntu-env
4775b2f378bb: Mounted from yungshenglu/ubuntu-env
883eafdbe580: Mounted from yungshenglu/ubuntu-env
19d043c86cbc: Mounted from yungshenglu/ubuntu-env
8823818c4748: Mounted from yungshenglu/ubuntu-env
latest: digest: sha256:eb3518265467d7014cb076015e415b343bf1b11a112ef64b02a209919b0f9d73 size: 7402
```

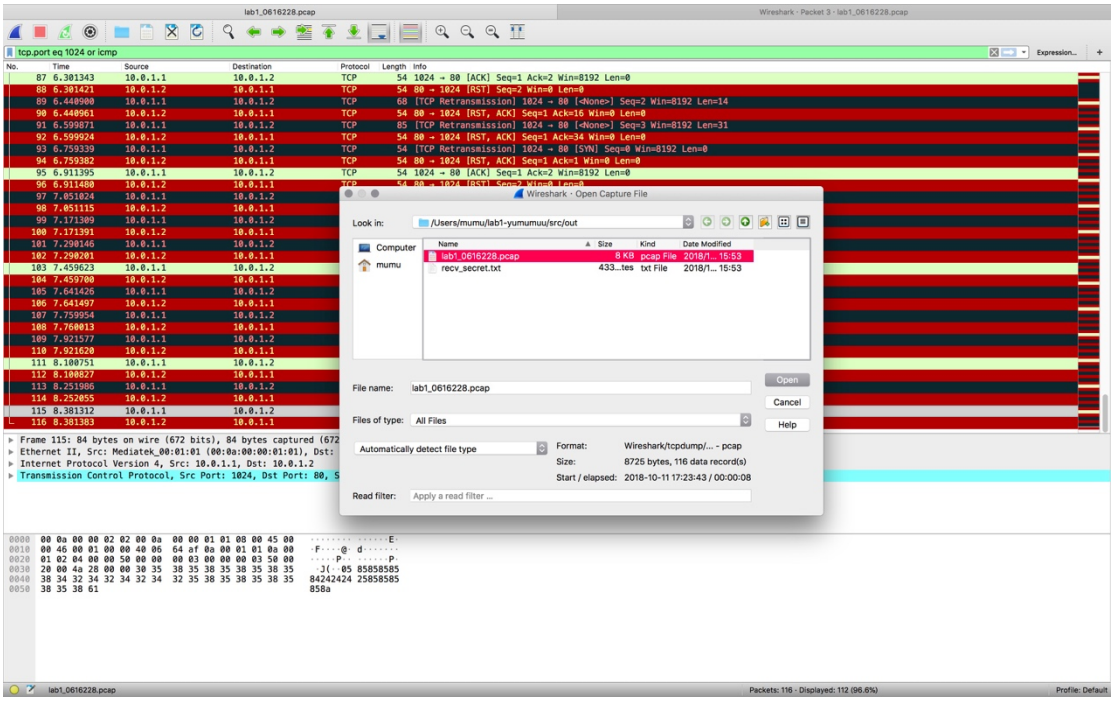
and github

```
de-MacBook-Pro:~ mumu$ git config --global user.name"yumumuu"
de-MacBook-Pro:~ mumu$ git config --global user.email"yusyuan.cs06@g2.nctu.edu.tw"
de-MacBook-Pro:~ mumu$ git add .
```

```
de-MacBook-Pro:Packet_Manipulation mumu$ git commit -m "Commit lab1 in class"
[master 7cca6ad] Commit lab1 in class
 2 files changed, 10 insertions(+), 6 deletions(-)
 create mode 160000 docker/Packet_Manipulation
de-MacBook-Pro:Packet_Manipulation mumu$ git remote set-url origin https://github.com/nctucn/lab1-yumumuu.git
de-MacBook-Pro:Packet_Manipulation mumu$ git push origin master
Counting objects: 4, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 769 bytes | 769.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/nctucn/lab1-yumumuu.git
   32f49c8..7cca6ad  master -> master
```

Task7 - Load PCAP via Wireshark

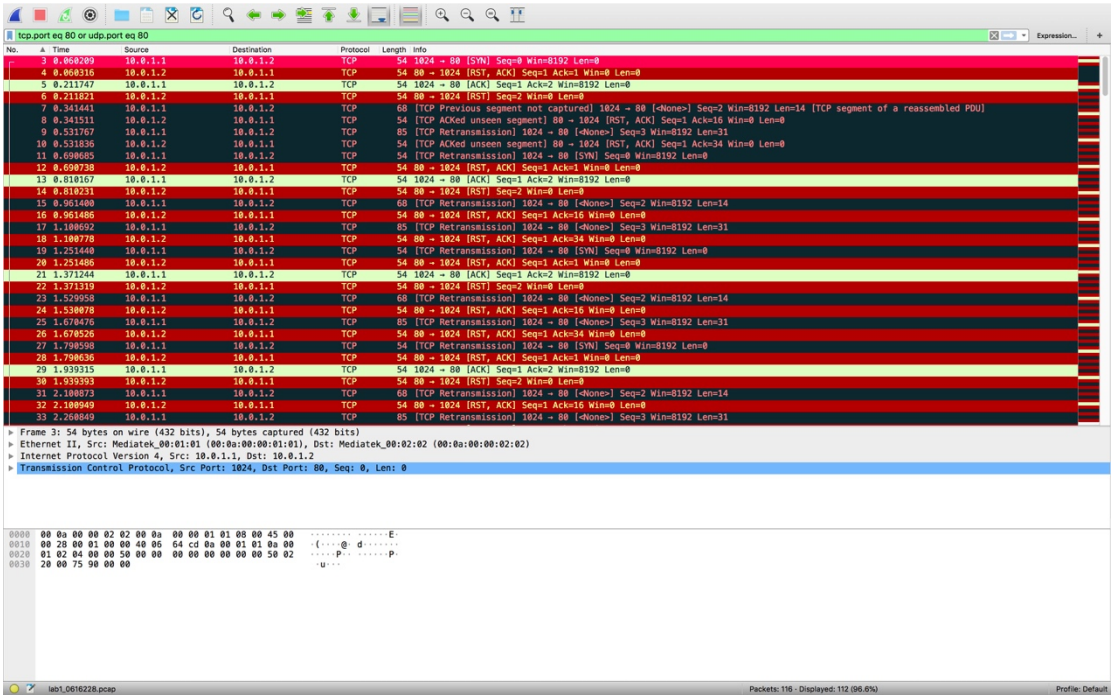
先用 git clone 下載我的檔案 然後用 wireshark 讀取 lab1_0616228.pcap



Task8 - Filter the target packet

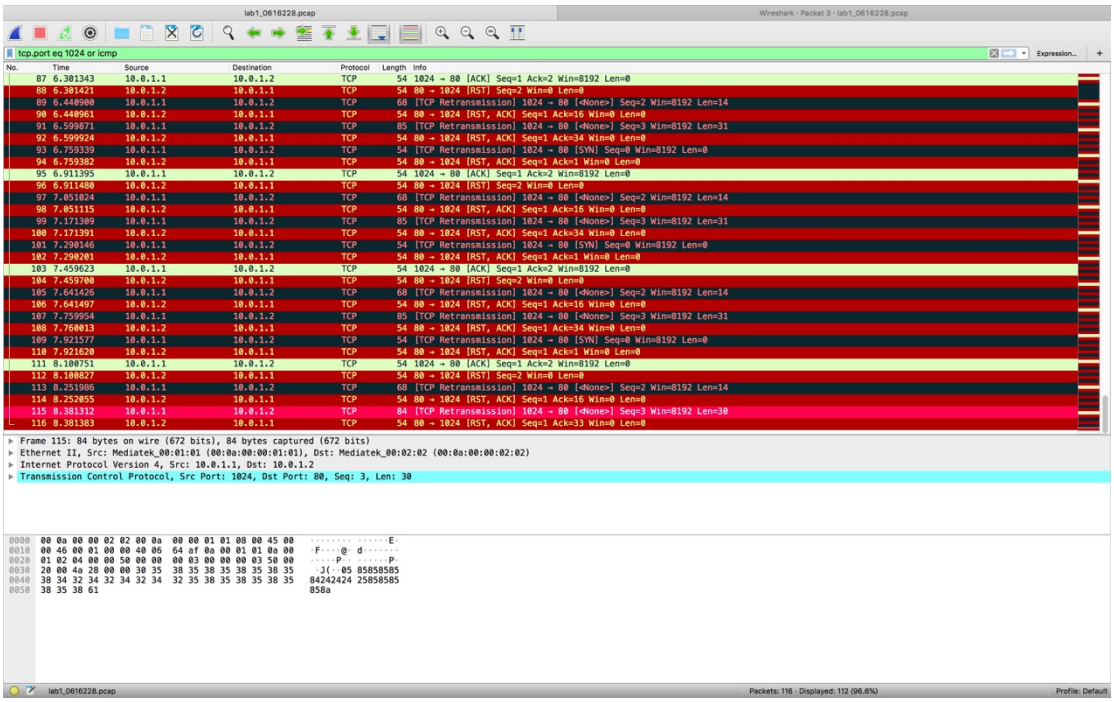
Filter the packets of our defined protocol

Commemd : tcp.port eq 80 or udp.port eq 80



Filter the packets with the “secret” bits

Commend : tcp.port eq 1024 or icmp



Find out the first digit of the “secret” payload in these packets and combine them as a 14-digit “secret” key

My Key : 82261608226160

Task9 - Decode the secret key

Input the secret key into ./src/decoder.py

```
de-MacBook-Pro:src mumu$ python decoder.py 82261608226160
[INFO] Your key is 82261608226160
[INFO] Decode successful
[INFO] Finish decoding
```

The output file is in ./src/out/lab1_0616228.png

Task10 - Bouns

What you have learned in this lab?

學到了一些 python 的寫法、tmux 的使用、wireshark 跟 docker 的使用、還有最重要的使用 github，如何建立環境，很酷的是在兩個終端機開 sender.py 和 receiver.py 傳送封包最後解碼得到 secret key，覺得很酷

What difficulty you have met in this lab?

誒，就是我一開始用 subline 改 dockerfile，然後一直卡在 task1 的 build environment 那裏，最後是用終端機 vim Dockerfile 才發現我沒有改到檔案，很氣，後來遇到的問題基本上都是沒有在正確的路徑輸入指令，改到正確的路徑之後就好了，希望下次的 lab 能夠更清楚說明路徑的部分。