NCTU CN2018 Lab. 1 – Packet Manipulation via Scapy

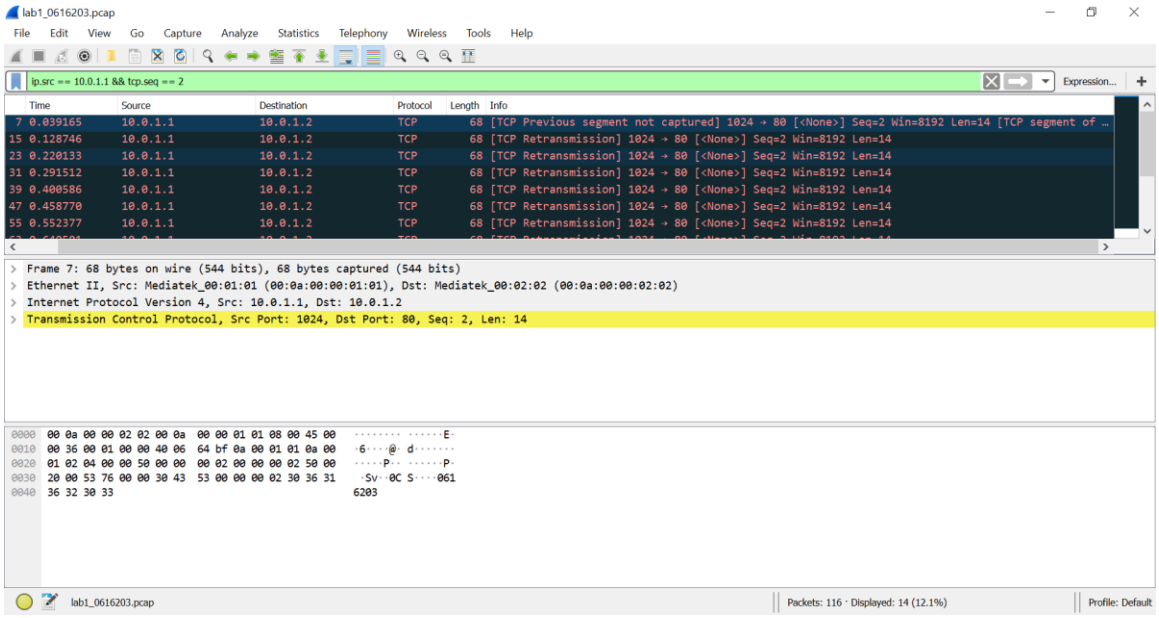
Student name: 林詩哲　Student ID: 0616203　Department: CS

Part A. Questions

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

```
ip.src == 10.0.1.1 && tcp.seq == 2
```

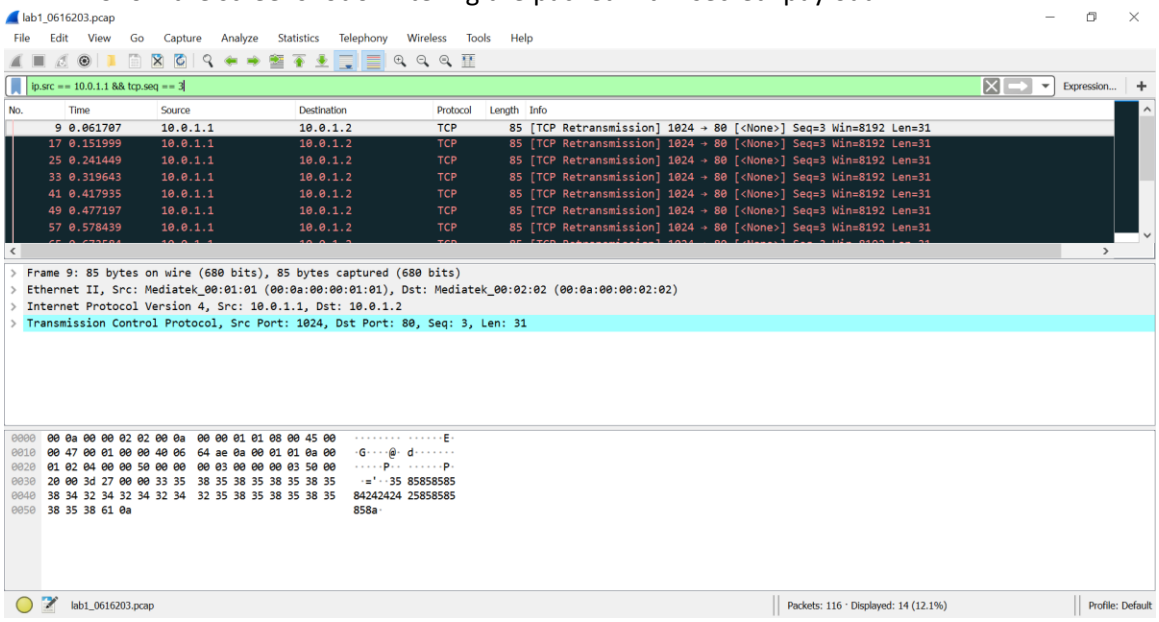
2. Show the screenshot of filtering the packet with customized header.



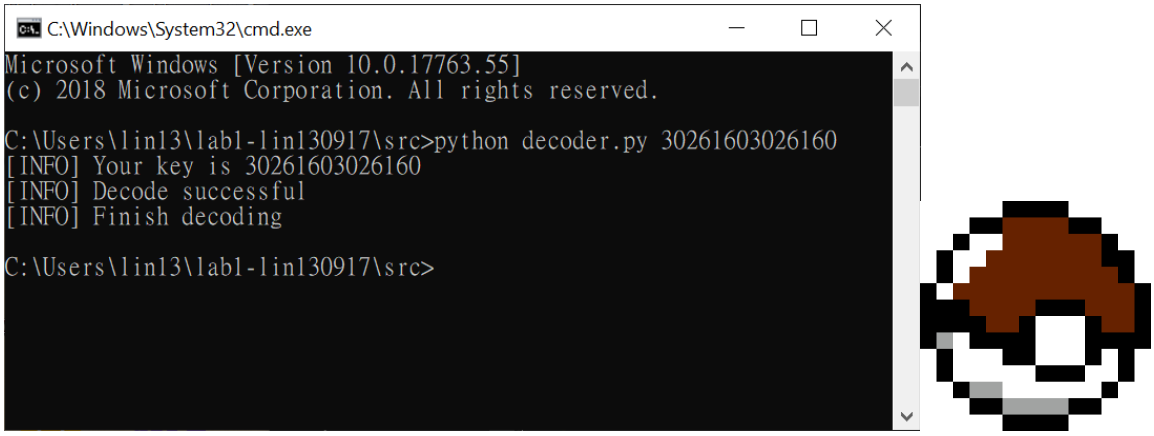
3. What is your command to filter the packet with “secret” payload on Wireshark?

```
ip.src == 10.0.1.1 && tcp.seq == 3
```

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

```
$ git clone https://github.com/yungshenglu/Packet_Manipulaton
```

• Copy the following configuration to the Dockerfile (./docker/Dockerfile)

```
# Download base image from yungshenglu/ubuntu-env:16.04
FROM yungshenglu/ubuntu-env:16.04
# Update software respository
RUN apt-get update
# Install software repository
RUN apt-get install -y tcpdump
# Install pip packages
RUN pip install scapy
# Set the container listens on the specified ports at runtime
EXPOSE 22
# Clone the repository from GitHub
RUN git clone https://github.com/yungshenglu/Packet_Manipulation.git
```

• Open the CMD (administration) and change the path to ./docker/ and build the environment as follows:

```
# Build the image from Dockerfile
docker build -t cn2018 .
# Build a container named cn2018_c from cn2018
docker run -d -p 9487:22 --privileged --name cn2018_c cn2018
# List port 22 mapping on cn2018_c
docker port cn2018_c 22
```

• Start the Docker container

• Open the PieTTY and connect to the Docker

```
Login: root
Password: cn2018
```

• Download required files from GitHub in the container

```
$ git clone https://github.com/yungshenglu/Packet_Manipulaton
```

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

```
# Create h2 network namespaces
ip netns add h2
# Delete h2 network namespaces
ip netns del h2
# Bring up the lookup interface in h2
ip netns exec h2 ip link set lo up
# Set the interface of h2 to h2-eth0
ip link set h2-eth0 netns h2
# Delete the interface of h2-eth0
ip link delete h2-eth0
# Activate h2-eth0 and assign IP address
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
# Disable all IPv6 on h2-eth0
ip netns exec h2 sysctl net.ipv6.conf.h2-eth0.disable_ipv6=1
# Set the gateway of h2 to 10.0.1.254
ip netns exec h2 ip route add default via 10.0.1.254
```

• Run main.sh to build the namespace

```
$ chmod +x main.sh
$ ./main.sh net
```

Task 2 – Define protocol via Scapy

• Define the protocol: ID header format. Copy the following code to ./src/Protocol.py

```
class Protocol(Packet):
# Set the name of protocol
name = 'Student'
# Define the fields in protocol
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),
]
```

Task 3 – Send packets

• Set the packet header in ./src/sender.py

```
# Set source and destination IP address
src_ip = '10.0.1.1'
dst_ip = '10.0.1.2'
# Set source and destination port
src_port = 1024
dst_port = 80
# Define IP header
ip = IP(src = src_ip, dst = dst_ip)
# Define customized header
my_id = '0616203'
my_dept = 'CS'
my_gender = 'male'
student = Protocol(id = my_id, dept = my_dept, gender = my_gender)
# TCP connection - ACK
ack = tcp_syn_ack.seq + 1
tcp_ack = TCP(sport = src_port, dport = dst_port, flags = 'A', seq = 1, ack = ack)
packet = ip / tcp_ack
send(packet)
print '[INFO] Send ACK'
# Send packet with customized header
ack = tcp_ack.seq + 1
tcp = TCP(sport = src_port, dport = dst_port, flags = '', seq = 2, ack = ack)
packet = ip / tcp / student
send(packet)
print '[INFO] Send packet with customized header'
# Send packet with secret payload
ack = tcp.seq + 1
tcp = TCP(sport = src_port, dport = dst_port, flags = '', seq = 3, ack = ack)
payload = Raw(secret[i])
packet = ip / tcp / payload
send(packet)
print '[INFO] Send packet with secret payload'
```

Task 4 – Sniff packets

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

```
# Set source IP address and destination interface
dst_iface = 'h2-eth0'
src_ip = '10.0.1.1'
# Sniff packets on destination interface
print '[INFO] Sniff on %s' % dst_iface
packets = sniff(iface = dst_iface, prn = lambda x: packetHandler(x))
# Dump the sniffed packet into PCAP file
print '[INFO] Write into PCAP file'
filename = './out/lab1_0' + id + '.pcap'
wrpcap(filename, packets)
```

Task 5 – Run sender and receiver

• Change the path to ./src/ and open tmux with horizontal two panes

```
# Open tmux
$ tmux
# Open new pane in horizontal
Ctrl-b
Shift-%
# Switch between two panes
Ctrl-b
Arrow-left/right key
```

• Run namespace h1 and h2

```
# Run namespace h1 in the left pane
$ ./scripts/main.sh run h1
# Run namespace h2 in the right pane
$ ./scripts/main.sh run h2
```

• Run sender and receiver

```
# Run receiver.py
h2> python receiver.py
# Run sender.py
h1> python sender.py
```

• Switch to right pane and press Ctrl-c to stop receiving and exit two namespace

```
Ctrl-c
h2> exit
h1> exit
```

• Use tcpdump to show the PCAP file

```
# Dump the PCAP via tcpdump
$ tcpdump -qns 0 -X -r <FILENAME>.pcap
```

Task 6 – Push all files to remote

• Open the CMD (administration) and push the image to Docker Hub

```
# Create a new image from a container’s changes
$ docker commit cn2018_c lin130917/cn2018_lab1
# Login to Docker registry
$ docker login
# Push an image to a registry
$ docker push lin130917/cn2018_lab1
```

• Push all files to GitHub

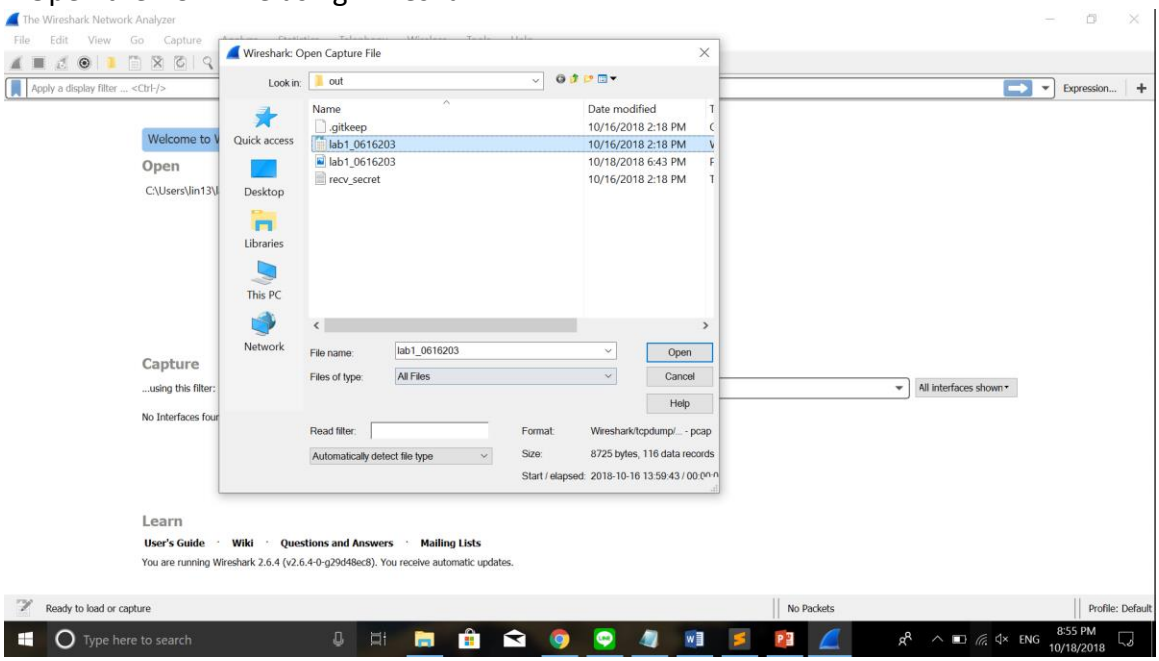
```
# Get and set repository or global options
$ git config --global user.name "Shizhe Lin"
$ git config --global user.email "lin130917.cs06@g2.nctu.edu.tw"
# Add all files into staging area
$ git add .
# Commit your files
$ git commit –m "Commit lab1 in class”
# Set the remote URL to my remote repository
$ git remote set-url origin https://github.com/nctucn/lab1-lin130917.git
# Push my files to remote repository
$ git push origin master
```

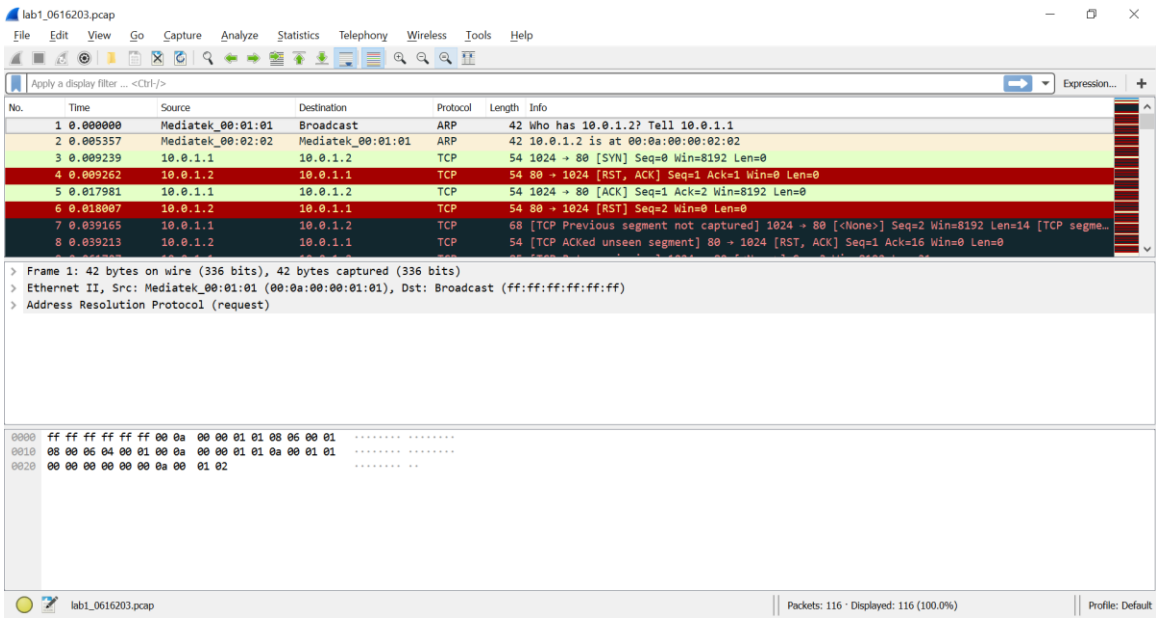
Task 7 – Load PCAP via Wireshark

- Download the code from GitHub

```
$ git clone https://github.com/nctucn/lab1-lin130917.git
```

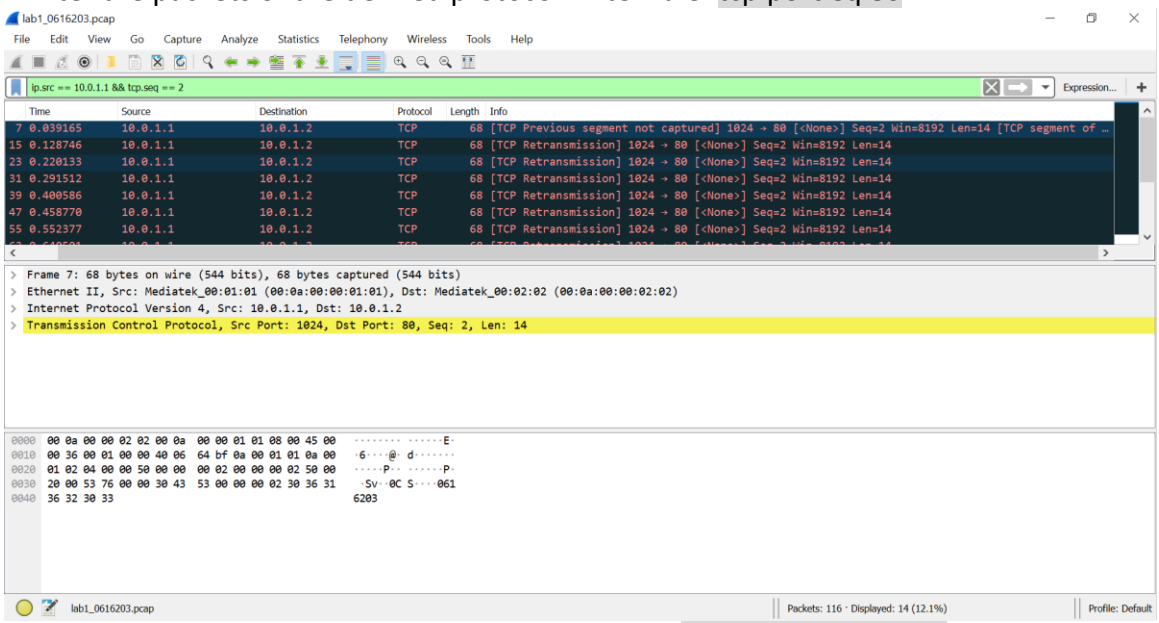
- Install Wireshark 2.6.3
- Open the PCAP file using Wireshark



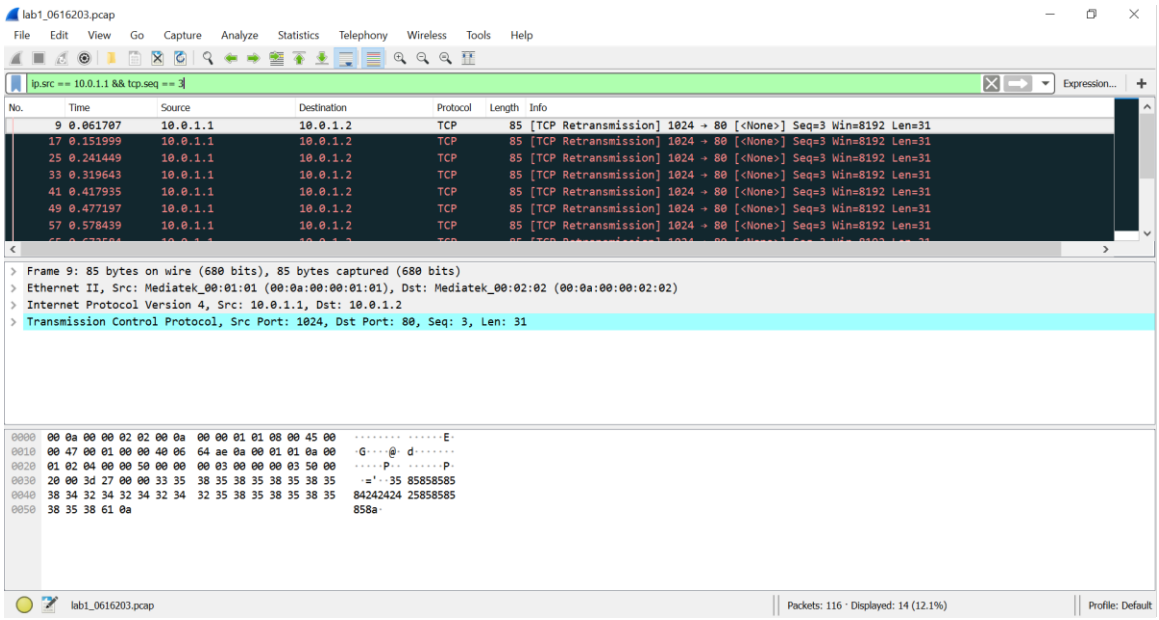


Task 8 – Filter the target packet

• Filter the packets of the defined protocol. Filter rule: tcp.port eq 80



• Filter the packets with the “secret” bits. Filter rule: tcp.port eq 80 or icmp

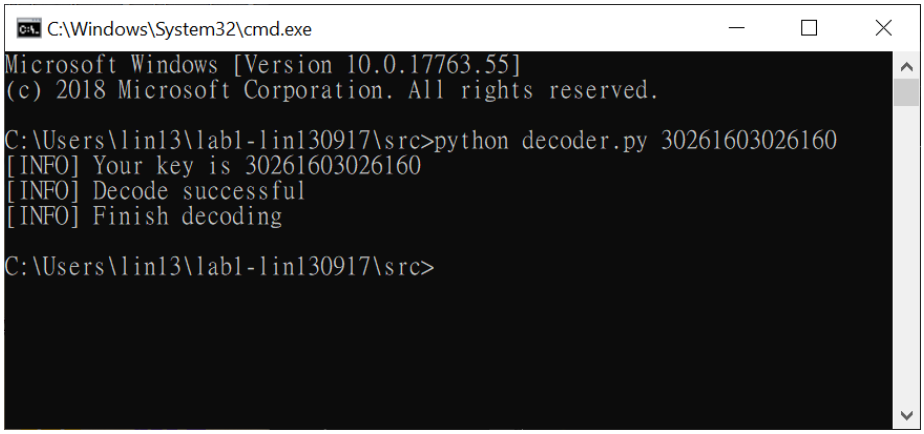


• Combine the first digit of the “secret” payload in these packets as a 14-digit “secret” key. My secret key is 30261603026160.

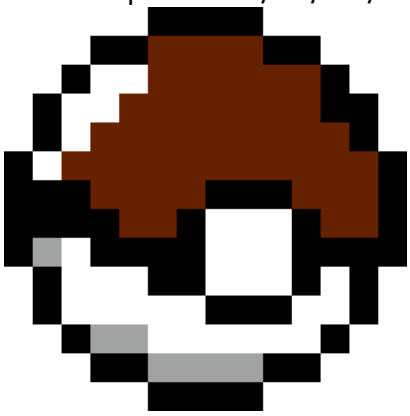
Task 9 – Decode the secret key

• Input the secret key into ./src/decoder.py on local machine

```
$ python decoder.py 30261603026160
```



• The output file is ./src/out/lab1_0616203.png



Part C. Bonus

• What you have learned in this lab?

首先在這次的lab我學到了一些工具的用法，像是git的基本指令以及如何把檔案push到github上，還有在Pietty使用vim以及tmux等等，再來就是這次的lab主要要我們學習的東西，先是建立好Docker container的環境，然後在container裡建好兩個host的namespace，再利用python的scapy套件從sender傳送封包到receiver，經由tcpdump擷取下來後，把pcap檔案拿到Wireshark指定一些規則過濾那些封包，取得一組secret key之後輸入到decoder裡，利用python的pillow套件解碼得到一張圖片。

• What difficulty you have met in this lab?

在這次的lab我遇到了一些困難，首先是不了解git的原理以及如何操作，花了一些時間才稍微了解git，再來就是那時候連不進去Docker container裡面，因為還不太了解Docker的運作方式，後來從知道要進去Docker的介面點進去container裡它會給你一組ip和port再輸入Pietty就可以連進去了，還有就是建立namespace的時候卡了很久，因為第一次建立namespace的時候因為code沒有打完整而沒有完全成功，回去修改main.sh再從新run一次namespace的時候terminal顯示錯誤h1已存在，之後才知道要用delns和dellink把它刪除之後再run一次就成功了。