

Computer Network – Project: Implementation of HTTP Sniffer

J201568008 김양선

- Introduction

+ code language: python3(>=3.4)

+ OS: Ubuntu 14.04

+ Library: Pcap(>= 0.11)

-Block

Function name	Function describe	Variables	Exception handle
main	User must choose Network device name in main function. Open file for write a HTTP header, pcap start live packet capture.	Input - argv	KeyboardInterrupt : It needs when stop this process by Keyboard Interrupt. It will saves data file.
payload_parser	Main function gives it a byte string. Then payload_parser function parsing a byte string to readable string list that split by (b'\r\n\r\n') and ('\r\n') after do this, encoding 'UTF-8'	Input - hdrpkt payload - file data_file - int idx output - a string data in file data_file return - int idx	UnicodeDecodeError : It needs when gets a image byte string.

- “Block by Block” Comments

```
import socket
from struct import unpack
import os
import pcap
import sys

def payload_parser(payload, data_file, idx):
    try:
        eth_length = 14
        eth = unpack('!6s6sH', payload[:eth_length])
        eth_protocol = socket.ntohs(eth[2])
```

```

if eth_protocol == 8:
    ip_header = payload[eth_length:20+eth_length]
    ip_header = unpack('!BBHHHBBH4s4s', ip_header)

```

eth_protocol ==8:
means IP Protocol.

```

version_ih = ip_header[0]
ip_header_length = (version_ih & 0xF) * 4
protocol = ip_header[6]
source_address = socket.inet_ntoa(ip_header[8])
dest_address = socket.inet_ntoa(ip_header[9])

```

Calculate its header size.

IP header has source address & destination adress

```

if protocol == 6:
    tcp_header_index = ip_header_length + eth_length
    tcp_header = payload[tcp_header_index:tcp_header_index+20]
    tcp_header = unpack('!HHLLBBHHH', tcp_header)

```

Protocol == 6:
Means TCP Protocol.
TCP Protocol is also IP Protocol's body.

```

source_port = tcp_header[0]
dest_port = tcp_header[1]
data_offset_reserved = tcp_header[4]
tcp_header_length = data_offset_reserved >> 4

```

It Splits header and body.

TCP's body is http header.

```

header_size = eth_length + ip_header_length + \
    tcp_header_length * 4

```

Header size will
calculated by byte.

```

data = payload[header_size:]
data_list = data.partition(b'\r\n\r\n')[0]

```

```

data_splited = str(data_list, 'utf-8').split('\r\n')
if "HTTP/1.1" in data_splited[0]:

```

Protocol == 6:
Means TCP
Protocol
It Splits header and body.
TCP's body is http header.
Header size will
calculated by byte.

```

    data_file.write("#" + str(idx) + " " +
        str(source_address) + ":" +
        str(source_port) + " " +
        str(dest_address) + ":" +
        str(dest_port) + "\n")

```

```

    idx = idx + 1
    for string in data_splited:
        data_file.write(string + "\n")

```

```

    data_file.write("\n")

```

```

except UnicodeDecodeError:
    pass

```

```
return idx
```

```
def main(argv):  
    print("Choose right device number")  
    devices = os.listdir('/sys/class/net/')  
  
    for index, device in enumerate(devices):  
        print("{0} {1}".format(index+1, device))  
  
    device_name = devices[int(input())-1]  
    captured = pcap.open_live(device_name, 65536, 1, 1)  
    idx = 0  
  
    (header, packet) = captured.next()  
    data_file = open("./result.txt", "w")  
    try:  
        while header is not None:  
            (header, packet) = captured.next()  
            idx = payload_parser(packet, data_file, idx)  
    except KeyboardInterrupt:  
        data_file.close()  
        pass  
  
if __name__ == "__main__":  
    main(sys.argv)
```

Read Network Interface's name.

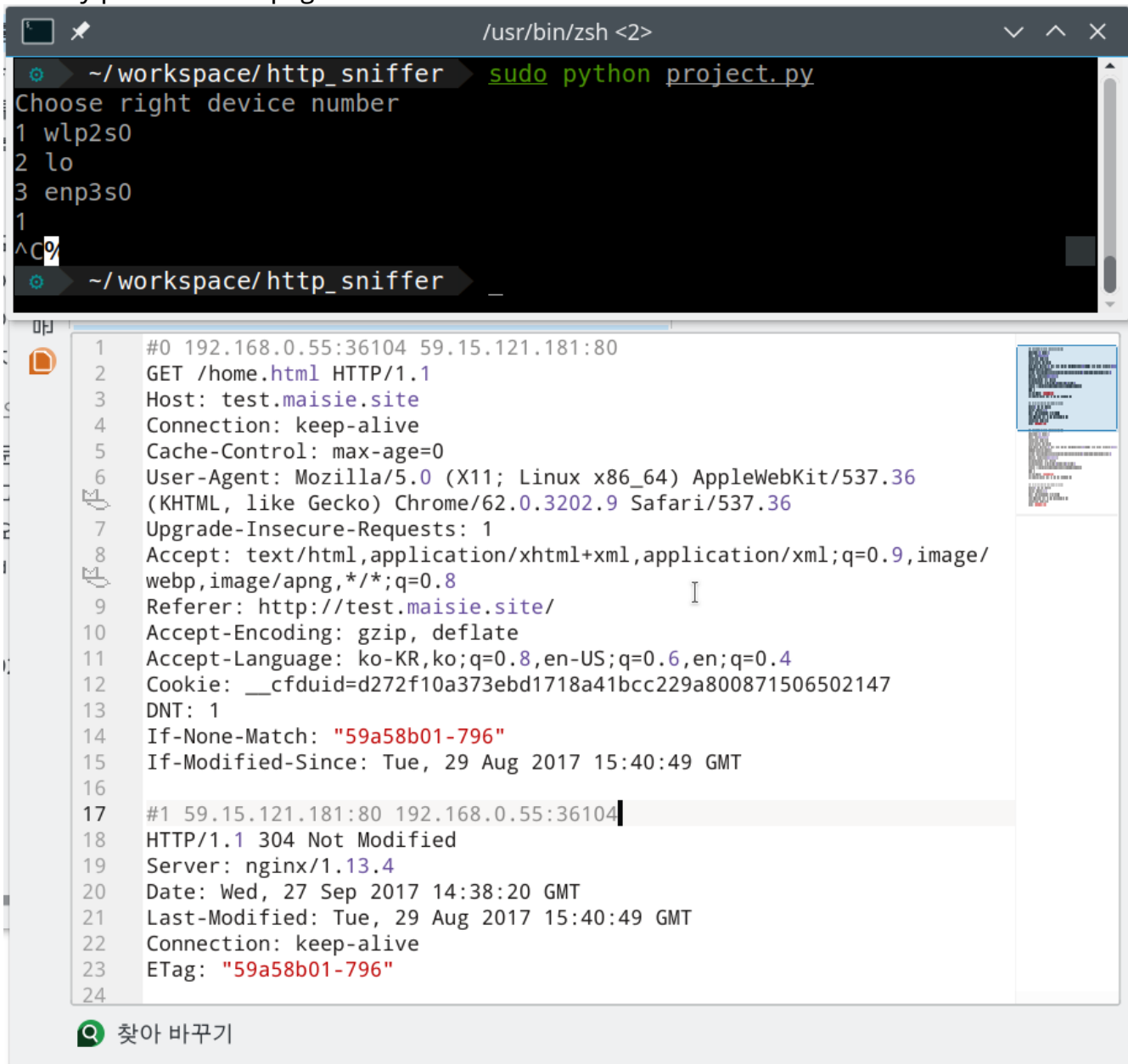
User can choose device that want to sniff.

Device captured from here.

Pcap.open_live needs device name, buffer byte, promisc, to_ms

- Screen Shots

+ in my personal homepage.



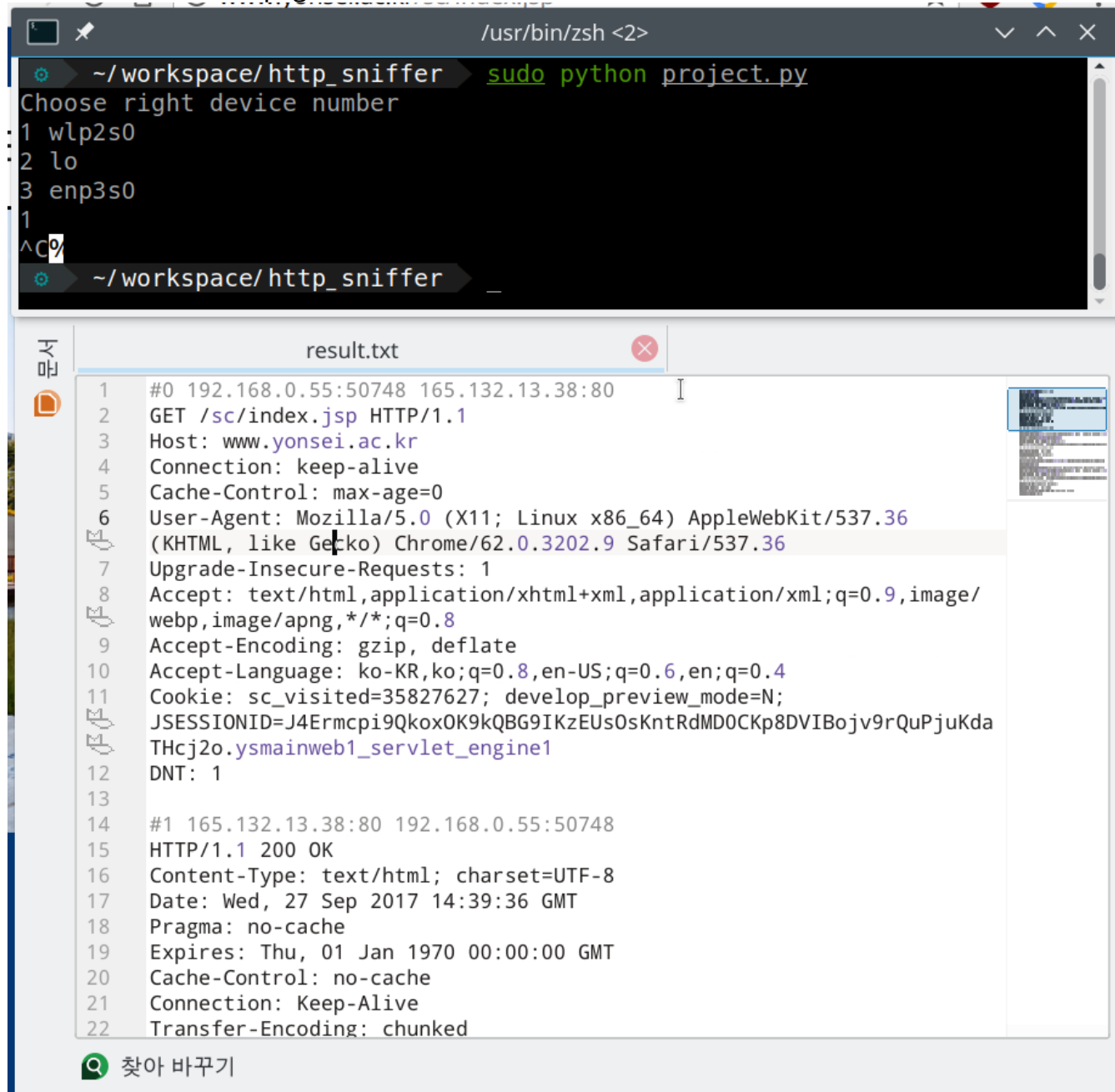
The image shows a terminal window and a web browser window. The terminal window is titled `/usr/bin/zsh <2>` and shows the execution of `sudo python project.py` in the directory `~/workspace/http_sniffer`. The program prompts the user to "Choose right device number" and lists three options: 1 wlp2s0, 2 lo, and 3 enp3s0. The user has selected option 1. The web browser window shows the HTTP request and response for the URL `http://test.maisie.site/`. The request is a GET request for `/home.html` with various headers including `User-Agent`, `Accept`, `Referer`, and `Cookie`. The response is a 304 Not Modified status with headers including `Server`, `Date`, `Last-Modified`, and `ETag`.

```
~/workspace/http_sniffer $ sudo python project.py
Choose right device number
1 wlp2s0
2 lo
3 enp3s0
1
^C
~/workspace/http_sniffer $
```

```
1 #0 192.168.0.55:36104 59.15.121.181:80
2 GET /home.html HTTP/1.1
3 Host: test.maisie.site
4 Connection: keep-alive
5 Cache-Control: max-age=0
6 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36
7 (KHTML, like Gecko) Chrome/62.0.3202.9 Safari/537.36
8 Upgrade-Insecure-Requests: 1
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/
10 webp,image/apng,*/*;q=0.8
11 Referer: http://test.maisie.site/
12 Accept-Encoding: gzip, deflate
13 Accept-Language: ko-KR,ko;q=0.8,en-US;q=0.6,en;q=0.4
14 Cookie: __cfduid=d272f10a373ebd1718a41bcc229a800871506502147
15 DNT: 1
16 If-None-Match: "59a58b01-796"
17 If-Modified-Since: Tue, 29 Aug 2017 15:40:49 GMT
18
19 #1 59.15.121.181:80 192.168.0.55:36104
20 HTTP/1.1 304 Not Modified
21 Server: nginx/1.13.4
22 Date: Wed, 27 Sep 2017 14:38:20 GMT
23 Last-Modified: Tue, 29 Aug 2017 15:40:49 GMT
24 Connection: keep-alive
25 ETag: "59a58b01-796"
```

찾아 바꾸기

+ in Yonsei univ. homepage.



The image shows a terminal window and a text editor. The terminal window, titled `/usr/bin/zsh <2>`, shows the execution of `sudo python project.py` in the directory `~/workspace/http_sniffer`. The program prompts the user to "Choose right device number" and lists three options: 1 wlp2s0, 2 lo, and 3 enp3s0. The user has entered '1'. The text editor, titled `result.txt`, displays the captured HTTP traffic. It shows a GET request from 192.168.0.55:50748 to 165.132.13.38:80 for the path `/sc/index.jsp`. The request headers include `Host: www.yonsei.ac.kr`, `Connection: keep-alive`, `Cache-Control: max-age=0`, `User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/62.0.3202.9 Safari/537.36`, `Upgrade-Insecure-Requests: 1`, `Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8`, `Accept-Encoding: gzip, deflate`, `Accept-Language: ko-KR;q=0.8,en-US;q=0.6,en;q=0.4`, `Cookie: sc_visited=35827627; develop_preview_mode=N; JSESSIONID=J4Ermcpi9QkoxOK9kQBG9IKzEU5OsKntRdMD0CKp8DVIBojv9rQuPjuKdaTHcj2o.ysmainweb1_servlet_engine1`, and `DNT: 1`. The response is an HTTP 200 OK from 165.132.13.38:80 to 192.168.0.55:50748, with headers including `Content-Type: text/html; charset=UTF-8`, `Date: Wed, 27 Sep 2017 14:39:36 GMT`, `Pragma: no-cache`, `Expires: Thu, 01 Jan 1970 00:00:00 GMT`, `Cache-Control: no-cache`, `Connection: Keep-Alive`, and `Transfer-Encoding: chunked`.

```
/usr/bin/zsh <2>
~/workspace/http_sniffer sudo python project.py
Choose right device number
1 wlp2s0
2 lo
3 enp3s0
1
^C%
~/workspace/http_sniffer _
```

```
result.txt
1 #0 192.168.0.55:50748 165.132.13.38:80
2 GET /sc/index.jsp HTTP/1.1
3 Host: www.yonsei.ac.kr
4 Connection: keep-alive
5 Cache-Control: max-age=0
6 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36
7 (KHTML, like Gecko) Chrome/62.0.3202.9 Safari/537.36
8 Upgrade-Insecure-Requests: 1
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/
10 webp,image/apng,*/*;q=0.8
11 Accept-Encoding: gzip, deflate
12 Accept-Language: ko-KR;q=0.8,en-US;q=0.6,en;q=0.4
13 Cookie: sc_visited=35827627; develop_preview_mode=N;
14 JSESSIONID=J4Ermcpi9QkoxOK9kQBG9IKzEU5OsKntRdMD0CKp8DVIBojv9rQuPjuKda
15 THcj2o.ysmainweb1_servlet_engine1
16 DNT: 1
17
18 #1 165.132.13.38:80 192.168.0.55:50748
19 HTTP/1.1 200 OK
20 Content-Type: text/html; charset=UTF-8
21 Date: Wed, 27 Sep 2017 14:39:36 GMT
22 Pragma: no-cache
23 Expires: Thu, 01 Jan 1970 00:00:00 GMT
24 Cache-Control: no-cache
25 Connection: Keep-Alive
26 Transfer-Encoding: chunked
```

찾아 바꾸기

- References

+ Official Documentation(included in pcap library file)

+ Pcap github (<https://github.com/CoreSecurity/pcapy>)

+ Mozilla Developer

Network(https://developer.mozilla.org/ko/docs/Web/HTTP/Basics_of_HTTP/MIME_types)