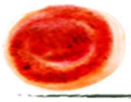
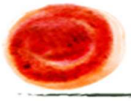


12. 패킷 캡처: scapy



SCAPY

scapy 소개

- 패킷 캡처, 만들기 등의 기능을 가진 모듈

```
python –m pip install –upgrade pip
pip install scapy
```

sniff() 함수

- 패킷을 캡처하는 함수

```
from scapy.all import *
sniff(prn, filter, store, iface, count, timeout)
```

prn – 패킷을 처리(출력)하는 함수
filter – 패킷을 거르는 내용
count – 캡처할 패킷 수(0=사용자가 중지할 때 까지)
store – 저장 여부를 결정
iface – 캡처할 인터페이스
timeout – 캡처할 시간(초 단위)

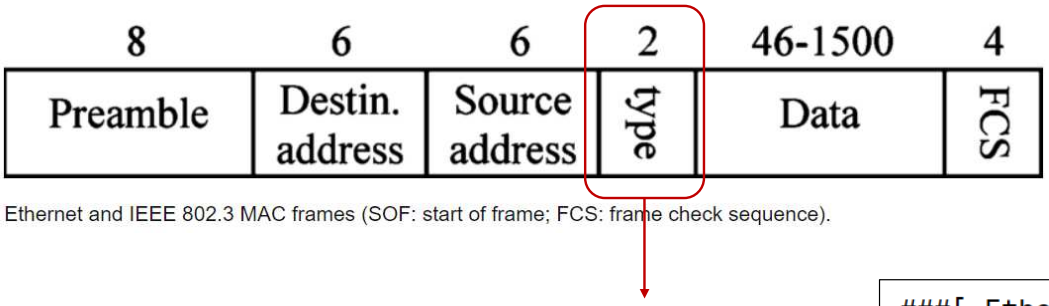
sniff() 함수를 이용한 패킷 캡처

```
from scapy.all import sniff

def pktshow(pkt):
    pkt.show()

def capture(n):
    sniff(prn=pktshow, count=n, filter='tcp')

if __name__ == '__main__':
    capture(2)
```

```
###[ Ethernet ]###
  dst       = 70:5d:cc:05:3c:54
  src       = a8:a1:59:1d:8f:42
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 69
     id        = 26907
     flags     = DF
     frag      = 0
     ttl       = 128
     proto     = tcp
     chksum    = 0x0
     src       = 192.168.0.35
     dst       = 52.111.232.4
     \options   \
###[ TCP ]###
        sport     = 51103
        dport     = https
        seq       = 2830486440
        ack       = 298538719
        dataofs   = 5
        reserved  = 0
        flags     = PA
        window    = 1022
        chksum    = 0xdd76
        urgptr    = 0
        options   = []
###[ Raw ]###
           load      = '\x17\x03\x03\x00
```

- 캡처한 패킷 각각에 대해 prn 함수 호출
- 캡처한 packet 객체를 이용 show() 함수는 패킷 내용 전체를 출력
- packet[0]는 네트워크 인트페이스 계층, packet[1]은 인터넷 계층, packet[2]는 전송계층에 접근할 수 있음.

Frame



Ethernet and IEEE 802.3 MAC frames (SOF: start of frame; FCS: frame check sequence).

0x0800 – IPv4

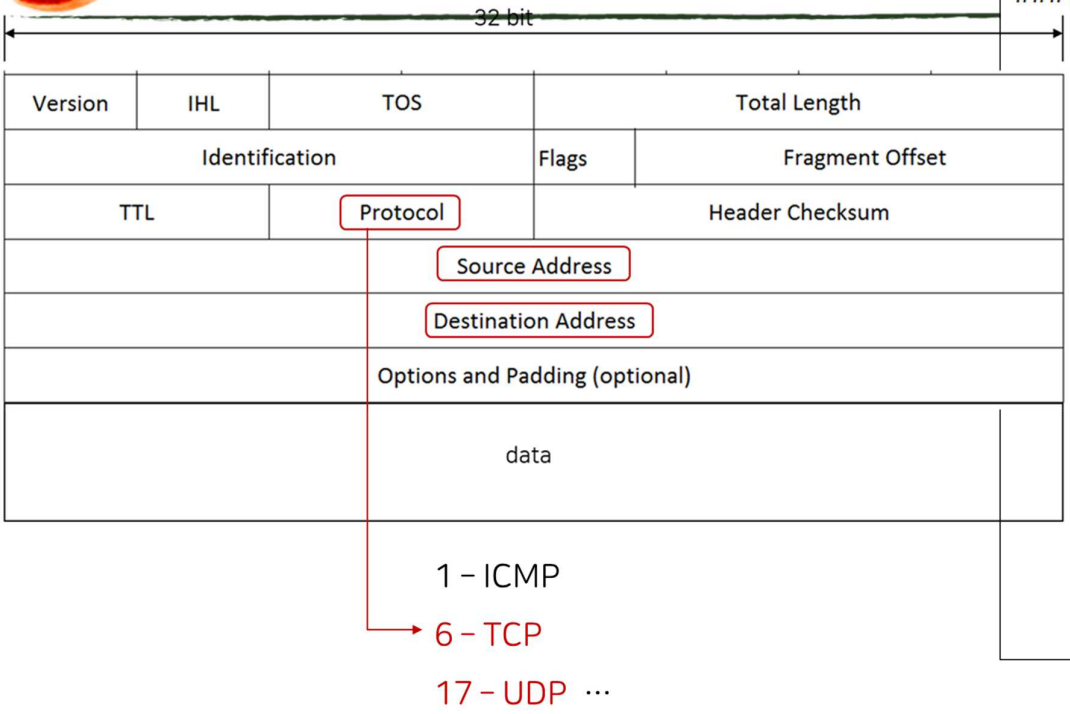
0x0806 – ARP

0x86DD – IPv6

…

```
###[ Ethernet ]###
  dst       = 70:5d:cc:05:3c:54
  src       = a8:a1:59:1d:8f:42
  type      = IPv4
```

IP Datagram



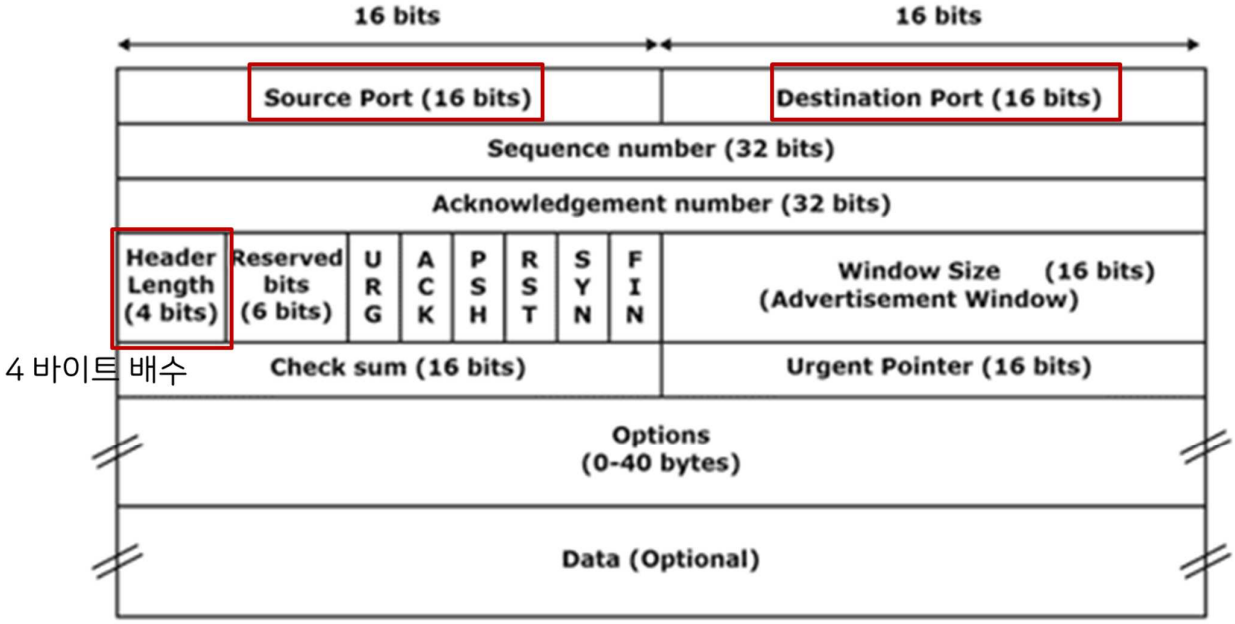
1 – ICMP

6 – TCP

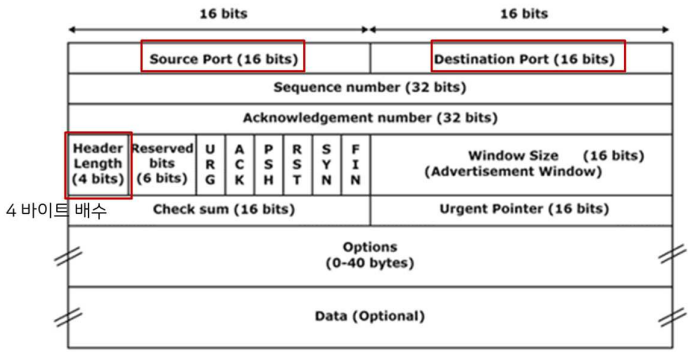
17 – UDP …

```
###[ IP ]###
  version   = 4
  ihl       = 5
  tos       = 0x0
  len       = 69
  id        = 26907
  flags     = DF
  frag      = 0
  ttl       = 128
  proto     = tcp
  chksum    = 0x0
  src       = 192.168.0.35
  dst       = 52.111.232.4
  \options   \
```

TCP Segment

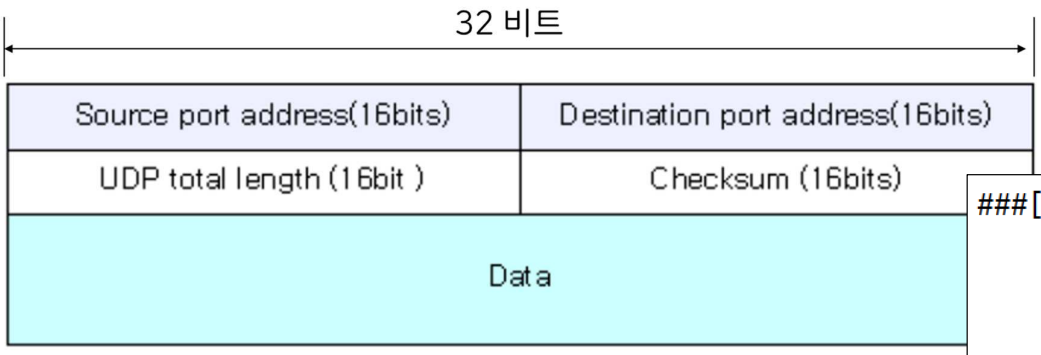


TCP Segment




```
###[ TCP ]###
        sport     = 51103
        dport     = https
        seq       = 2830486440
        ack       = 298538719
        dataofs   = 5
        reserved  = 0
        flags     = PA
        window    = 1022
        chksum    = 0xdd76
        urgptr    = 0
        options   = []
###[ Raw ]###
           load      = '\x17\x03\x0
```

UDP 패킷




```
###[ UDP ]###
        sport     = 60629
        dport     = 51007
        len       = 152
        chksum    = 0x831c
###[ Raw ]###
           load      = "J\\xb5\
```

캡처한 패킷에서 정보 추출

- 인덱스 이용하여 프로토콜 계층 정보에 접근
  - 0 : network interface layer
  - 1 : internet layer
  - 2 : transport layer
  - 사용자 데이터 : transport layer의 load
- 각 계층의 정보 접근 형식

  packet[idx].identifier

  예: packet[0].dst

```
###[ Ethernet ]###
  dst       = 70:5d:cc:05:3c:54
  src       = a8:a1:59:1d:8f:42
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 69
     id        = 26907
     flags     = DF
     frag      = 0
     ttl       = 128
     proto     = tcp
     chksum    = 0x0
     src       = 192.168.0.35
     dst       = 52.111.232.4
     \options   \
###[ TCP ]###
        sport     = 51103
        dport     = https
        seq       = 2830486440
        ack       = 298538719
        dataofs   = 5
        reserved  = 0
        flags     = PA
        window    = 1022
        chksum    = 0xdd76
        urgptr    = 0
        options   = []
###[ Raw ]###
           load      = '\x17\x03\x03\x00
```

캡처한 패킷에서 정보 추출

- 패킷의 getlayer() 함수 이용

  packet.getlayer('Ether')

  packet.getlayer('IP')

  packet.getlayer('TCP')

  packet.getlayer('Raw')

  …

```
###[ Ethernet ]###
  dst       = 70:5d:cc:05:3c:54
  src       = a8:a1:59:1d:8f:42
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 69
     id        = 26907
     flags     = DF
     frag      = 0
     ttl       = 128
     proto     = tcp
     chksum    = 0x0
     src       = 192.168.0.35
     dst       = 52.111.232.4
     \options   \
###[ TCP ]###
        sport     = 51103
        dport     = https
        seq       = 2830486440
        ack       = 298538719
        dataofs   = 5
        reserved  = 0
        flags     = PA
        window    = 1022
        chksum    = 0xdd76
        urgptr    = 0
        options   = []
###[ Raw ]###
           load      = '\x17\x03\x03\x00
```

sniffing filter

- 캡처할 패킷을 위한 조건

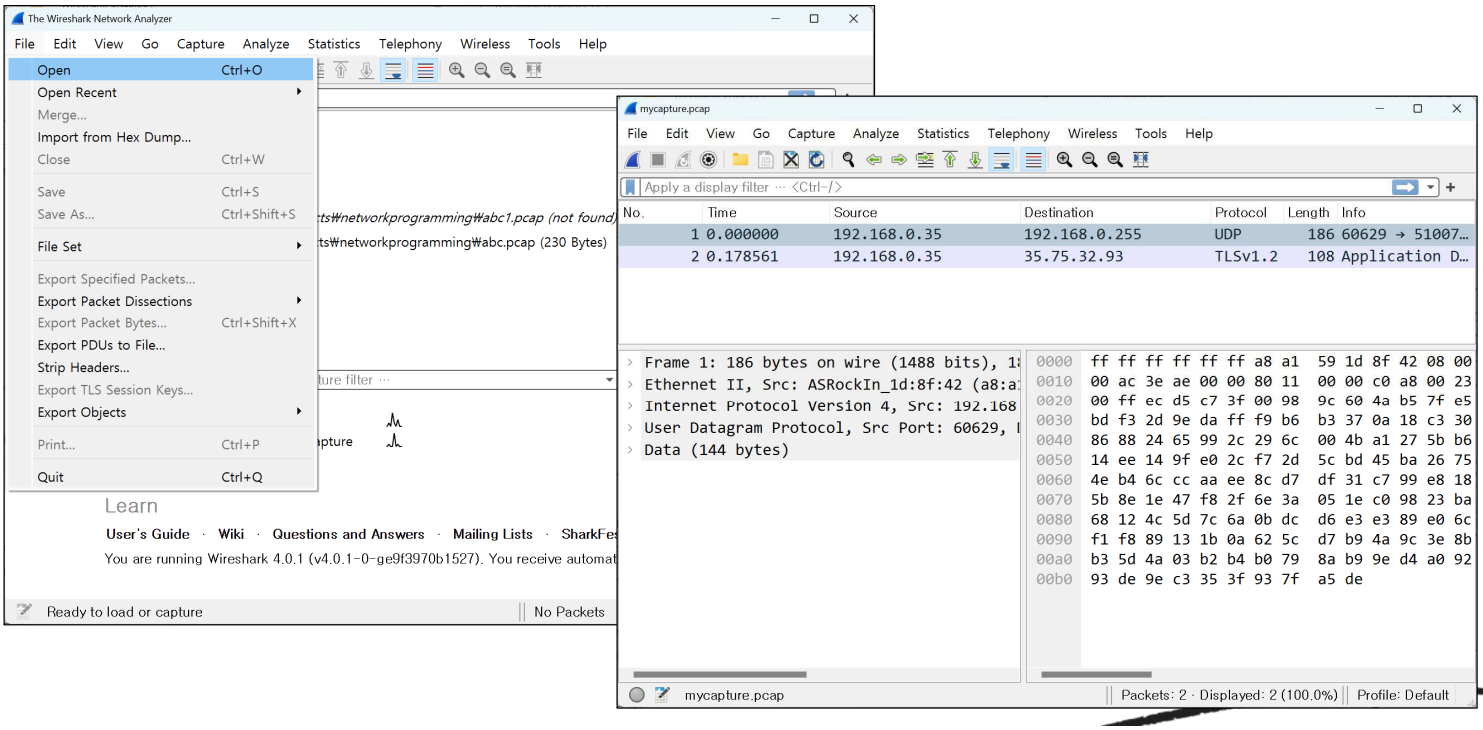
  [protocol] [port #]

캡처한 패킷 저장

- 캡처한 패킷을 저장하여 wireshark 등에서 확인

```
cpkt = sniff(prn=pktshow, count=n, store=True)
wrpcap('mycapture.pcap', cpkt)
```

저장한 패킷 정보 확인 ; wireshark 이용



UDP 패킷 캡처 – UDP 서버

```
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.bind(('', 9999))
sock.settimeout(5)
count = 0
try:
    while True:
        try:
            data, addr = sock.recvfrom(1024)
            while True:
                print(data.decode(), count)
                sock.sendto(struct.pack('>is', count,b'₩n'), addr)
                count += 1
                time.sleep(5)
        except socket.timeout:
            print('timeout')
except KeyboardInterrupt:
    pass
sock.close()
```

UDP 패킷 캡처 – UDP 클라이언트

```
import socket
import struct
count = 0
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.sendto('good'.encode(), ('localhost', 9999))
sock.settimeout(3)
while True:
    try:
        buf, _ = sock.recvfrom(1024)
        if not buf:
            break
        print(f'{sock} : {struct.unpack(">i", buf[:4])}')
        count += 1
    except socket.timeout:
        pass

sock.close()
```

UDP 패킷 캡처 – UDP 캡처

```
from scapy.all import sniff, get_if_list, get_if_addr
import struct
from scapy.config import conf

def pktanalysis(pkt):
    udp = pkt.getlayer('UDP')
    udp.show()
    raw = pkt.getlayer('Raw')
    if raw:
        print(struct.unpack('>i',raw.load[:4]))
```

```
def pktcapture():
    sniff_iface = None
    print(conf.iface)
    print(get_if_addr(conf.iface))

    ifaces = get_if_list()
    for iface in ifaces:
        if 'loopback' in iface.lower():
            sniff_iface = iface
            break
    print(get_if_addr(sniff_iface))
    sniff(prn=pktanalysis, filter='udp port 9999',
iface=sniff_iface)
if __name__ == '__main__':
    pktcapture()
```

패킷 만들기(2.4.5버전 기준)

패킷 만들기

- TCP/IP 프로토콜 계층/프로토콜의 클래스 이용
- 네트워크 인터페이스 계층 : Ether()
- 인터넷 계층 : IP(), ARP()
- 전송 계층 : TCP(), UDP()
- 응용 계층 : Raw()
- 간단한 예:

```
pkt = Ethernet()/IP()/TCP()/Raw()
```

Ethernet()

- 네트워크 인터페이스를 위한 클래스
- 속성
  - dst : 패킷을 수신할 MAC 주소
  - src : 자신의 MAC 주소
  - type : 종류(default: 0x9000 – loopback)

```
from scapy.layers.inet import Ether

Ether().show()
```

```
###[ Ethernet ]###
  dst       = ff:ff:ff:ff:ff:ff
  src       = [redacted]
  type      = 0x9000
```

IP()

- TCP/IP 인터넷 계층의 IP 프로토콜을 위한 클래스
- 속성:
  - src : 패킷을 전송하는 IP 주소
  - dst : 패킷을 수신하는 IP 주소

```
from scapy.layers.inet import IP

IP().show()
```

```
###[ IP ]###
  version   = 4
  ihl       = None
  tos       = 0x0
  len       = None
  id        = 1
  flags     =
  frag      = 0
  ttl       = 64
  proto     = ip
  chksum    = None
  src       = 127.0.0.1
  dst       = 127.0.0.1
  \options   \
```

ARP()

- ARP 프로토콜을 위한 클래스
- 속성 :

```
###[ ARP ]###
  hwtype    = 0x1
  ptype     = IPv4
  hwlen     = None
  plen      = None
  op        = who-has
  hwsrc     = [redacted]
  psrc      = 192.168.0.35
  hwdst     = 00:00:00:00:00:00
  pdst      = 0.0.0.0
```

```
from scapy.layers.l2 import ARP

ARP().show()
```

ICMP()

- ICMP 프로토콜을 위한 클래스
- 속성 :

```
###[ ICMP ]###
  type      = echo-request
  code      = 0
  chksum    = None
  id        = 0x0
  seq       = 0x0
  unused    = ''
```

```
from scapy.layers.inet import ICMP

ICMP().show()
```

TCP()

- TCP 프로토콜을 위한 클래스
- 속성 :

```
###[ TCP ]###
  sport     = ftp_data
  dport     = http
  seq       = 0
  ack       = 0
  dataofs   = None
  reserved  = 0
  flags     = S
  window    = 8192
  chksum    = None
  urgptr    = 0
  options   = ''
```

```
from scapy.layers.inet import TCP

TCP().show()
```

Raw()

- 형식 없는 바이트 열을 위한 클래스
- 보통 응용 계층의 데이터
- 속성 :

```
###[ Raw ]###
  load       = ''
```

```
from scapy.packet import Raw

Raw().show()
```

ARP 패킷 만들고 전송

- ARP 프로토콜을 IP 주소를 이용하여 MAC 결정
  - '홍길동' 몇 번이니?(IP주소)
  - 1번 입니다.(MAC 주소)
- 같은 subnet에 있는 시스템의 MAC 주소 질의
  - 공유기, 라우터 너머에 있는 시스템 MAC 주소 알 필요 X

ARP 패킷 만들고 전송의 예:

```
from scapy.layers.l2 import ARP
from scapy.layers.inet import sr1
import argparse

parser = argparse.ArgumentParser(prog='gethw.py')
parser.add_argument('-i', '--ip', required=True)
args = parser.parse_args()

pkt = ARP(pdst=args.ip)
pkt.show()
result = sr1(pkt)
if result :
   result.show()
   print(result.hwdst)
   print(result.getlayer("ARP").hwdst)
```

ICMP 패킷 만들고 전송

- ICMP 프로토콜은 타겟이 동작 중인지 확인 목적

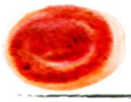
```
from scapy.layers.inet import sr1, IP, ICMP
import argparse

parser = argparse.ArgumentParser(prog='ping.py')
parser.add_argument('-i', '--ip', required=True)
args = parser.parse_args()

pkt = IP(dst=args.ip)/ICMP()
pkt.show()

result = sr1(pkt)
if result :
   result.show()
```

패킷 생성과 캡처

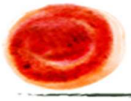
```
from scapy.all import sniff
import sys
from scapy.layers.inet import IP, ICMP, sr1
import threading

def pktanalysys(pkt):
    pkt.show()

def capture(*cnt):
    sniff(prn=pktanalysys, count=cnt[0], filter='icmp')
```

패킷 생성과 캡처

```
def pkicmp():
    print('Sending ICMP Packet')
    pkt = IP(dst=sys.argv[1])/ICMP()
    result = sr1(pkt)
    if result:
        result.show()

if __name__ == '__main__':
    cap_th = threading.Thread(target=capture, args=(5,))
    cap_th.start()
    snd_th = threading.Thread(target=pkicmp)
    snd_th.start()
    cap_th.join()
    snd_th.join()
```



Q&A



source code
    tab

이것은 이것은

이것은 이것은