Network Fundamental

강사. 유효곤

(ugonfor@gmail.com)

본 수업의 강의자료는 이경문(gilgil, BoB멘토, 고려대 사이버국방학과 겸임교수)멘토님의 강의자료를 상당수 차용하였음을 밝힙니다.

Contents

- 정보보안 윤리
- 네트워크란,
  - Network Model
  - Ethernet header, IP header, TCP/UDP header
  - Network Security
- 네트워크 프로그래밍
  - Socket Programming
  - Pcap Programming

정보보안 윤리

- 정보보호전문가로 성장하고, 전문기술을 습득하는 데 있어서 윤리의식은 가장 중요한 것 중 하나이다.
  - 윤리의식의 위에 기술을 쌓아야만 정보보안 전문가로 성장할 수 있다.
- 해킹은 합법이 아니다.
  - 정보보안 기술을 배우면, 기술을 시험해보고 싶다.
  - 하지만, 허가되지 않은 해킹은 **법적 처벌을 받을 수 있다**
  - 블랙햇으로 활동하면 절대 돌아오지 못함
- **착하게 해킹하자**
- 특히 **네트워크 기술 공공장소**에서 시험해보지 마세요!

네트워크란,

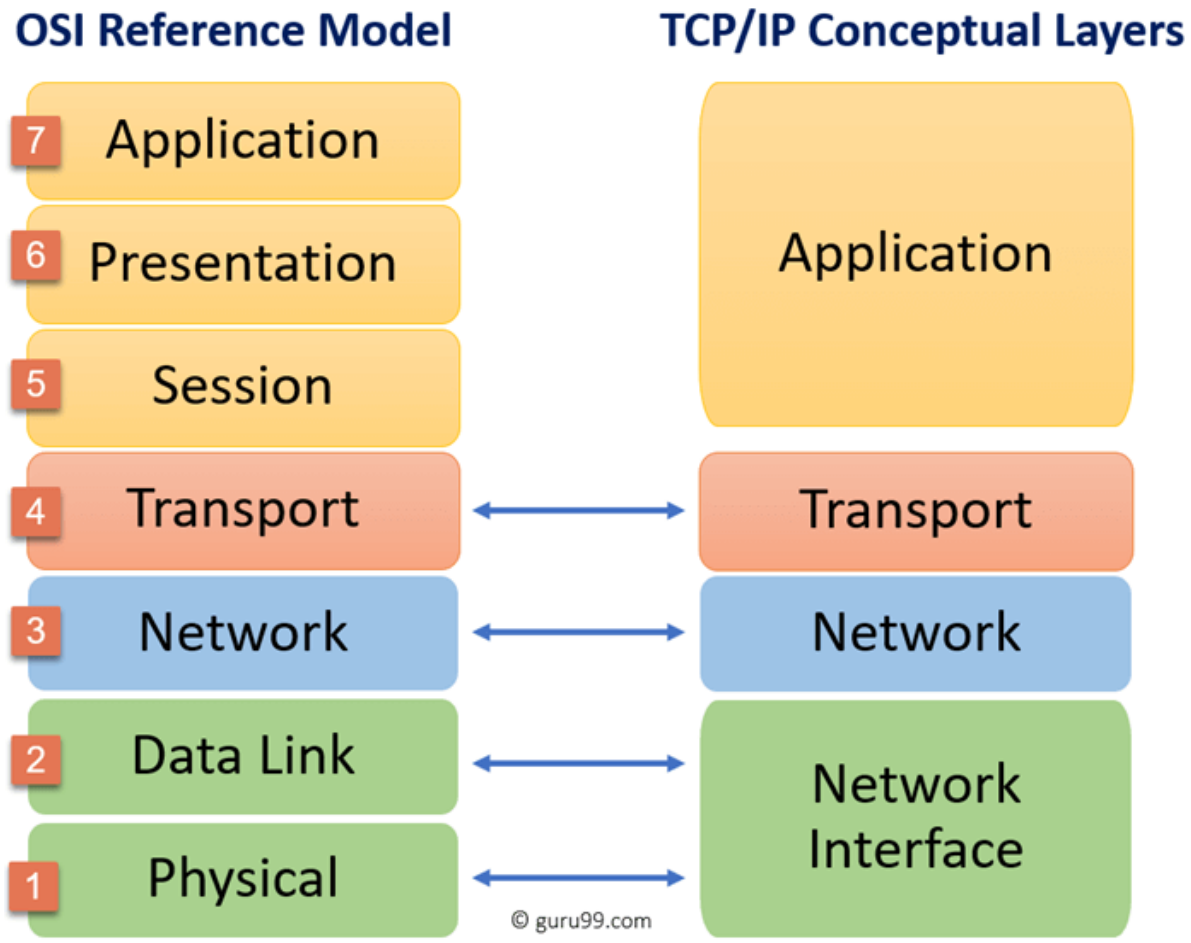
Network Model

Ethernet header, IP header, TCP/UDP header

Network Security

Network Model

OSI reference Model / TCP.IP Model



Network Model

OSI reference Model / TCP.IP Model

OSI reference Model	TCP/IP Model	Layer Number	Protocol
Application	Application	L7(Firewall)	HTTP, SSH, Telnet, ···
Presentation			
Session			
Transport	Transport	L4(NAT)	TCP, UDP
Network	Network	L3(Router)	IP, IPv6, ARP, ICMP, ···
Data Link	Network Interface	L2(Switch)	Ethernet
Physical			

Network Model

OSI reference Model / TCP.IP Model

OSI reference Model	TCP/IP Model	Layer Number	Protocol
Application	Application	L7(Firewall)	HTTP, SSH, Telnet, ···
Presentation			
Session			
Transport	Transport	L4(NAT)	TCP, UDP
Network	Network	L3(Router)	IP, IPv6, ARP, ICMP, ···
Data Link	Network Interface	L2(Switch)	Ethernet
Physical			

Network Model

OSI reference Model / TCP.IP Model

- 왜 Layer를 나눌까?
- Layer를 나누지 않았다면 어떤 일이 발생할 것인가?

Network Model

OSI reference Model / TCP.IP Model

- 왜 Layer를 나눌까?
- Layer를 나누지 않았다면 어떤 일이 발생할 것인가?
  - Layer를 나누지 않았다면, 어플리케이션 개발자가 모든 네트워크 통신에 대해 고려해주어야 한다.
  - 새로운 네트워크 통신 규약이 생기게 되면 다시 처음부터 모든 것을 고려해주어야 한다.
- Layer를 나누어 두었기 때문에, HTTP, FTP, DNS 등의 Application layer의 프로토콜이 새로 생성될 때, IP와 MAC 주소 기반으로 기기를 찾을 수 있는 것

Network Model

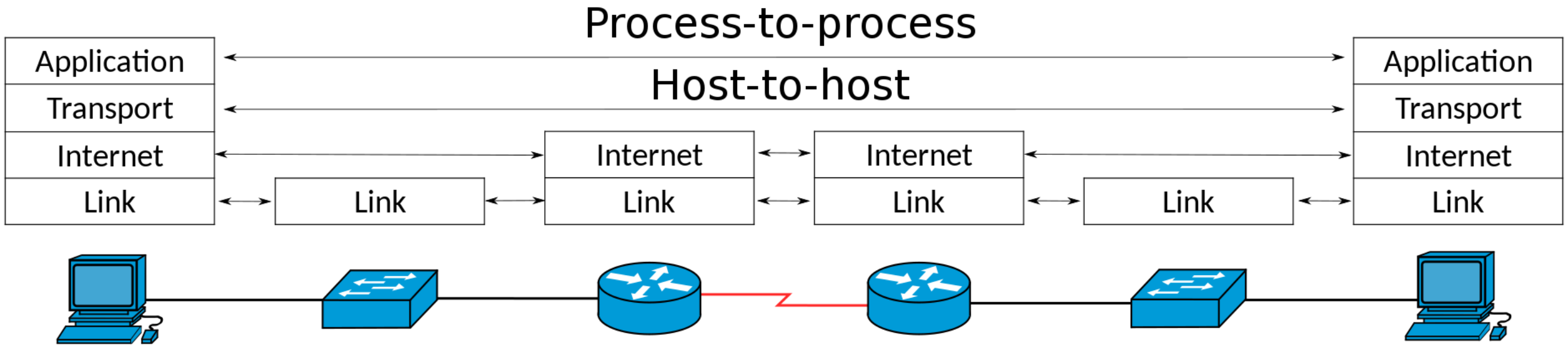
OSI reference Model / TCP.IP Model

- 왜 Layer를 나눌까?
- Layer를 나누지 않았다면 어떤 일이 발생할 것인가?
  - Layer를 나누지 않았다면, 어플리케이션 개발자가 모든 네트워크 통신에 대해 고려해주어야 한다.
  - 새로운 네트워크 통신 규약이 생기게 되면 다시 처음부터 모든 것을 고려해주어야 한다.
- Layer를 나누어 두었기 때문에, HTTP, FTP, DNS 등의 Application layer의 프로토콜이 새로 생성될 때, IP와 MAC 주소 기반으로 기기를 찾을 수 있는 것
- 통신규약을 따르고 있기에, 핸드폰의 종류가 많고, 이를 개발하는 개발자들이 모두 다름에도 서로 통신을 오류 없이 해낼 수 있다.

Network Model

네트워크 설계의 철학

- End-to-End Principle
  - 네트워크를 설계하는 데 있어서, 어플리케이션의 특정 기능들은 모두 End 에서 처리하고자 하는 철학
  - 네트워크의 중간에서 패킷 수, 제어 등을 하지 않고, 모두 종단(end point)에서 해결 하자!
  - 네트워크의 중간에서 제어를 하게 되면, delay가 생기게 됨
    - (그럴 바에 다시 한번 패킷을 보내자!, 상위 레이어에서 처리하자!)



Ethernet, IP, TCP/UDP header

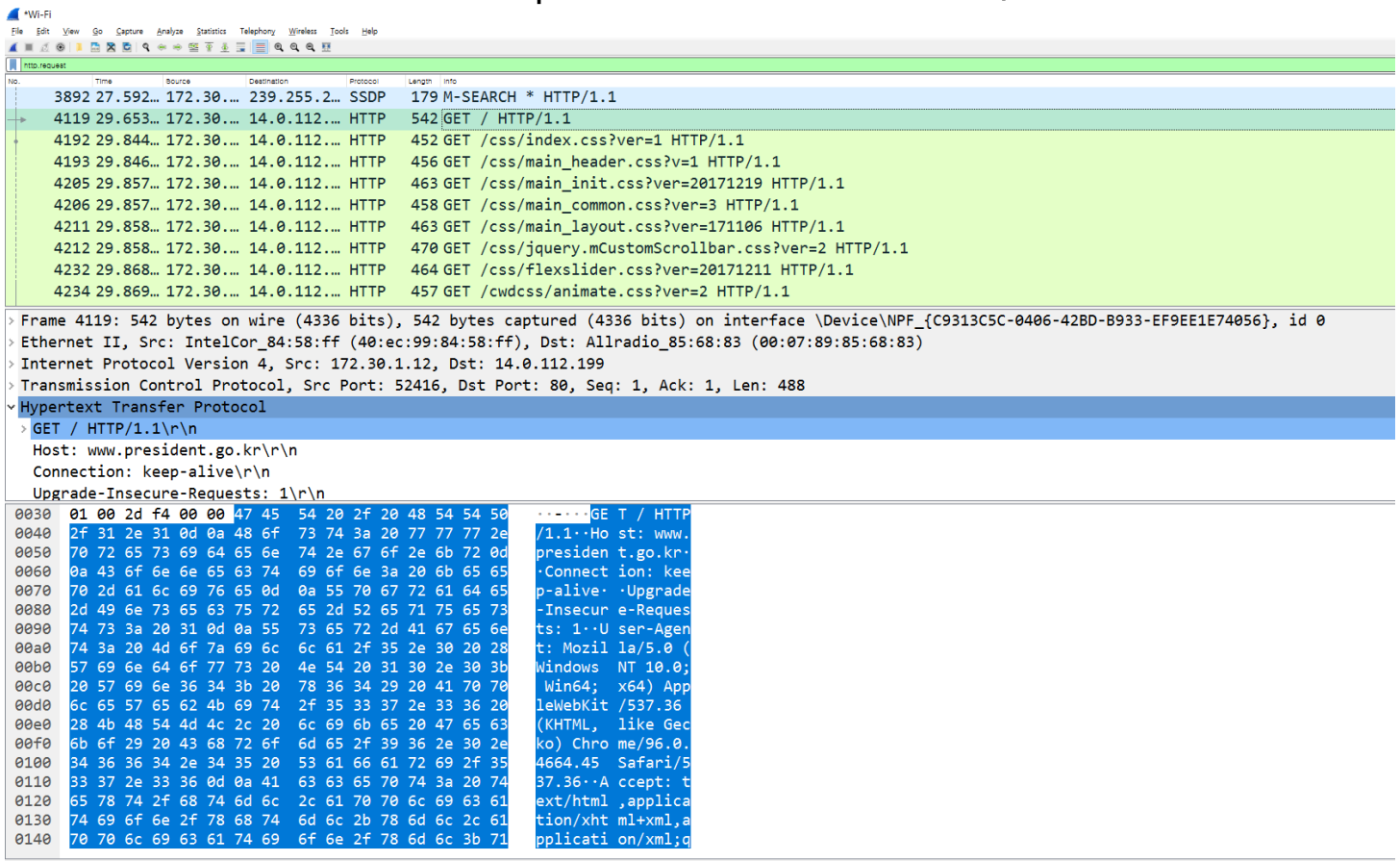
Packet capture

- 대표적으로 웹사이트를 접속하게 되면 Ethernet, IP, TCP, HTTP 프로토콜들을 기반으로 통신하게 됨.
- 각각의 패킷(네트워크 통신의 가장 작은 단위)은 각 프로토콜에 따라 데이터들이 감싸져 있다.

Ethernet, IP, TCP/UDP header

Packet capture

- Wireshark(https://www.wireshark.org/) 이용하며 패킷을 직접 확인해보자.
- http://www.president.go.kr/ (놀랍게도 청와대 사이트는 https를 지원하지 않는다.)
- Network byte order에 조심할 것



Ethernet, IP, TCP/UDP header

Packet capture

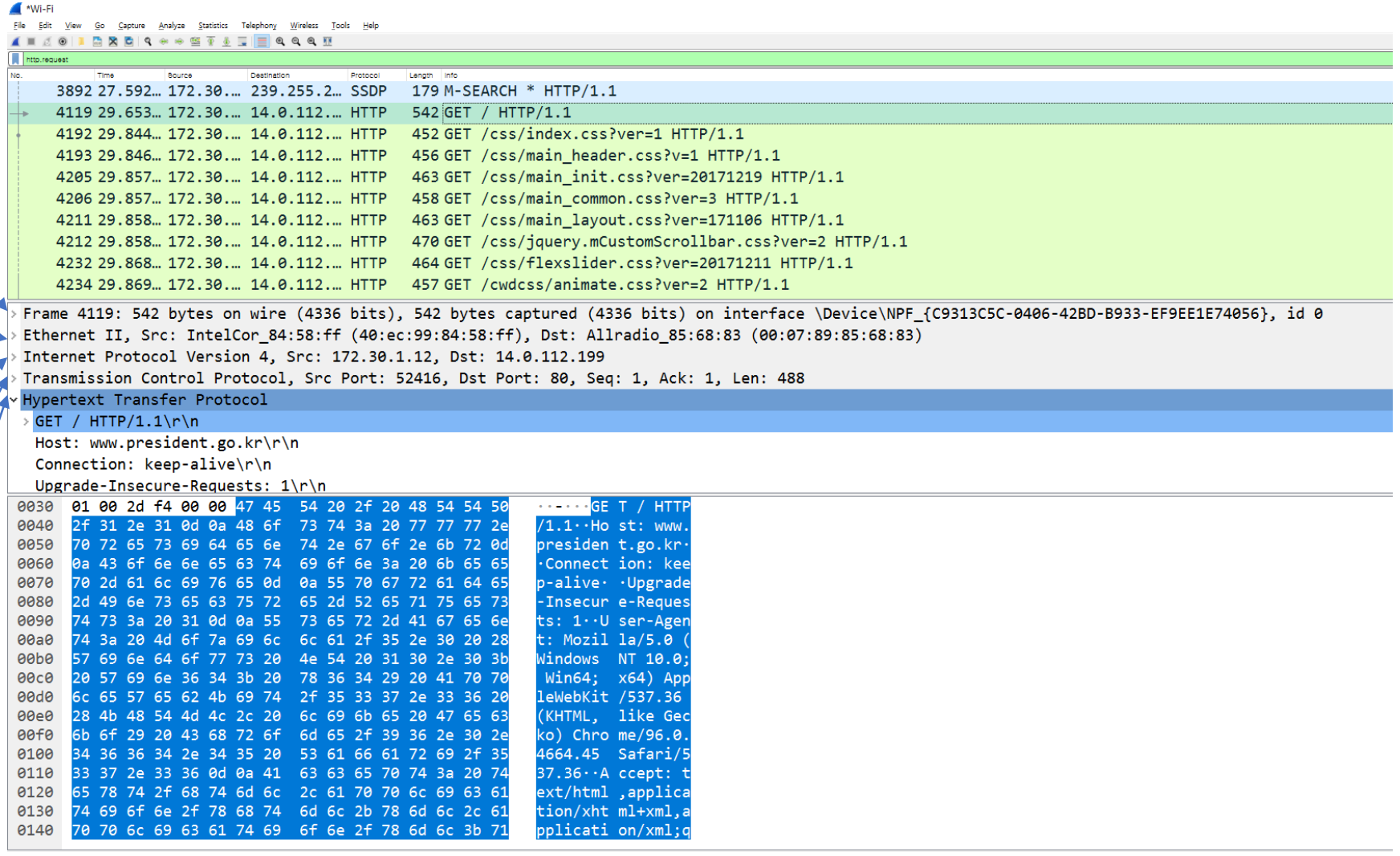
Frame

Ethernet

IP

TCP

HTTP



Ethernet, IP, TCP/UDP header

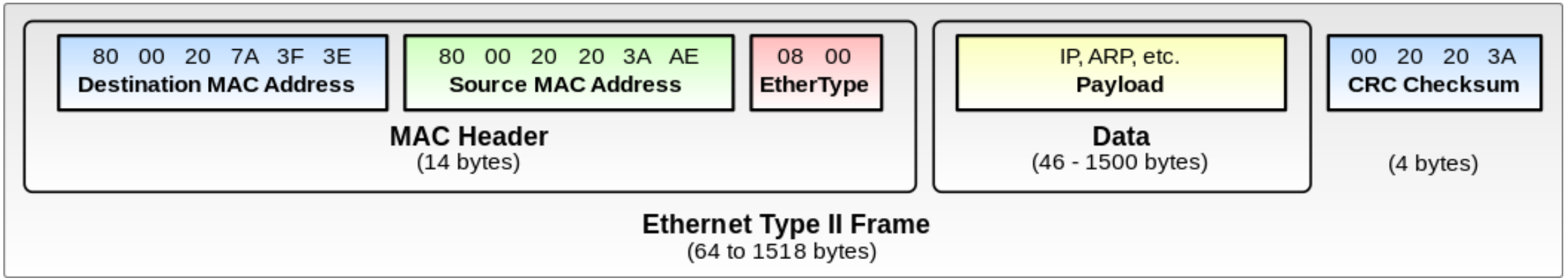
모든 프로토콜을 알아야 하는 가?

- Ethernet, IP, TCP/UDP 는 필수
- 나머지 프로토콜들은 필요에 따라 그때그때 배우면 됨.

프로토콜	목적
Ethernet	MAC 주소 정보를 통해 근거리 통신의 목적지 확인
IP	IP 주소 정보를 통해 장거리 통신의 목적지 확인
UDP	포트 정보를 통해, 목적지 기기의 포트 정보 확인
TCP	포트 정보를 통해, 목적지 기기의 포트 정보 확인 UDP에서 혼잡제어, 흐름제어, 데이터 유실 등을 고려한 프로토콜

Ethernet, IP, TCP/UDP header

Ethernet



https://en.wikipedia.org/wiki/Ethernet_frame

Ethernet, IP, TCP/UDP header

IP(Internet Protocol)

IPv4 header format

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Version				IHL				DSCP						ECN		Total Length															
4	32	Identification																Flags			Fragment Offset											
8	64	Time To Live								Protocol								Header Checksum															
12	96	Source IP Address																															
16	128	Destination IP Address																															
20	160	Options (if IHL > 5)																															
⋮	⋮																																
56	448																																

https://en.wikipedia.org/wiki/IPv4

Ethernet, IP, TCP/UDP header

IP(Internet Protocol)

IPv4 header format

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	IP 버전				헤더 길이/4				TOS(서비스 타입)								패킷 길이(Byte)															
4	32	패킷이 여러 개로 쪼개진 경우를 위한 id																Fragment Flag				쪼개진 조각의 오프셋											
8	64	패킷 생존시간								프로토콜 종류								헤더 체크썸															
12	96	Source IP Address																															
16	128	Destination IP Address																															
20	160	Options (if IHL > 5)																															
⋮	⋮																																
56	448																																

https://en.wikipedia.org/wiki/IPv4

Ethernet, IP, TCP/UDP header

UDP(User Datagram Protocol)

UDP datagram header

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Source port																Destination port															
4	32	Length																Checksum															

https://en.wikipedia.org/wiki/User_Datagram_Protocol

Ethernet, IP, TCP/UDP header

TCP(User Datagram Protocol)

TCP segment header

Offsets	Octet	0								1								2								3							
Octet	Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	0	Source port																Destination port															
4	32	Sequence number																															
8	64	Acknowledgment number (if ACK set)																															
12	96	Data offset				Reserved 0 0 0				NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size														
16	128	Checksum																	Urgent pointer (if URG set)													
20	160	Options (if *data offset* > 5. Padded at the end with "0" bytes if necessary.)																															
⋮	⋮																																
60	480																																

https://en.wikipedia.org/wiki/Transmission_Control_Protocol

Ethernet, IP, TCP/UDP header

TCP(User Datagram Protocol)

TCP segment header

Offsets	Octet	0								1								2								3							
Octet	Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	0	Source port																Destination port															
4	32	Sequence number 내가 몇 번째로 보내는 패킷인 것인지																															
8	64	Acknowledgment number (if ACK set) 몇번째 SEQ를 잘 받았는 지																															
12	96	Data offset TCP 헤더 길이				Reserved 0 0 0 사용안함				NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size 여유 패킷 수용 가능															
16	128	Checksum																	Urgent pointer (if URG set) 긴급히 처리해야하는 데이터의 마지막 바이트 위치														
20	160	Options (if *data offset* > 5. Padded at the end with "0" bytes if necessary.)																															
⋮	⋮																																
60	480																																

https://en.wikipedia.org/wiki/Transmission_Control_Protocol

Ethernet, IP, TCP/UDP header

TCP(User Datagram Protocol)

Flag	기능
NS	혼잡제어 관련 Flag
CWR	혼잡제어 관련 Flag
ECE	혼잡제어 관련 Flag
URG	긴급 패킷(우선순위가 높음)
ACK	TCP 데이터를 수신하였음을 알려주는 bit
PSH	가능한 빨리 L7에 데이터 전달할 것
RST	강제 연결 종료(리셋)
SYN	연결 시작
FIN	종료

https://en.wikipedia.org/wiki/Transmission_Control_Protocol

Network Security

네트워크 보안 기법

- 보안기법
  - 방화벽(Firewall)
  - 침입탐지시스템(IDS)
  - 유해사이트 차단
  - ...
- 해킹 기법
  - MITM(Man in the middle)
  - Sniffing
  - DDOS
  - ...

Network Security

네트워크 보안 기법

- 대부분의 경우 패킷을 모니터링하다가 허가/ 차단을 결정한다.
- 실제로 서비스를 하게 되면, 몇 백만개의 패킷이 돌아다니기에 성능이 굉장히 좋아야 한다.
  - Resource의 효율을 최대화 시켜야 함
  - **알고리즘이 굉장히 중요한 분야**
- 어쩔 수 없이 네트워크의 중간에서 패킷을 제어하게 되는 데, End-to-end principle에 따라 패킷을 제어하지 않는 서비스가 보통 더 선호 됨.
- Ex, 유해사이트 차단

Network Security

네트워크 해킹 기법

- MITM(Man in the middle)
  - A와 B사이 통신의 중간에서 공격자가 내용을 도청하거나 조작하는 공격 기법
- Sniffing
  - A와 B 통신의 내용을 공격자가 도청하는 방법
  - Wireshark도 패킷 스니핑을 해주는 툴 중 하나임.
- DDOS류
  - 엄청나게 많은 패킷을 특정 기기에 보내서, 시스템의 리소스에 과부하를 주는 공격 기법
  - 이를 위해 IDS나 방화벽이 필수!

Network Programming

Socket Programming

Pcap programming

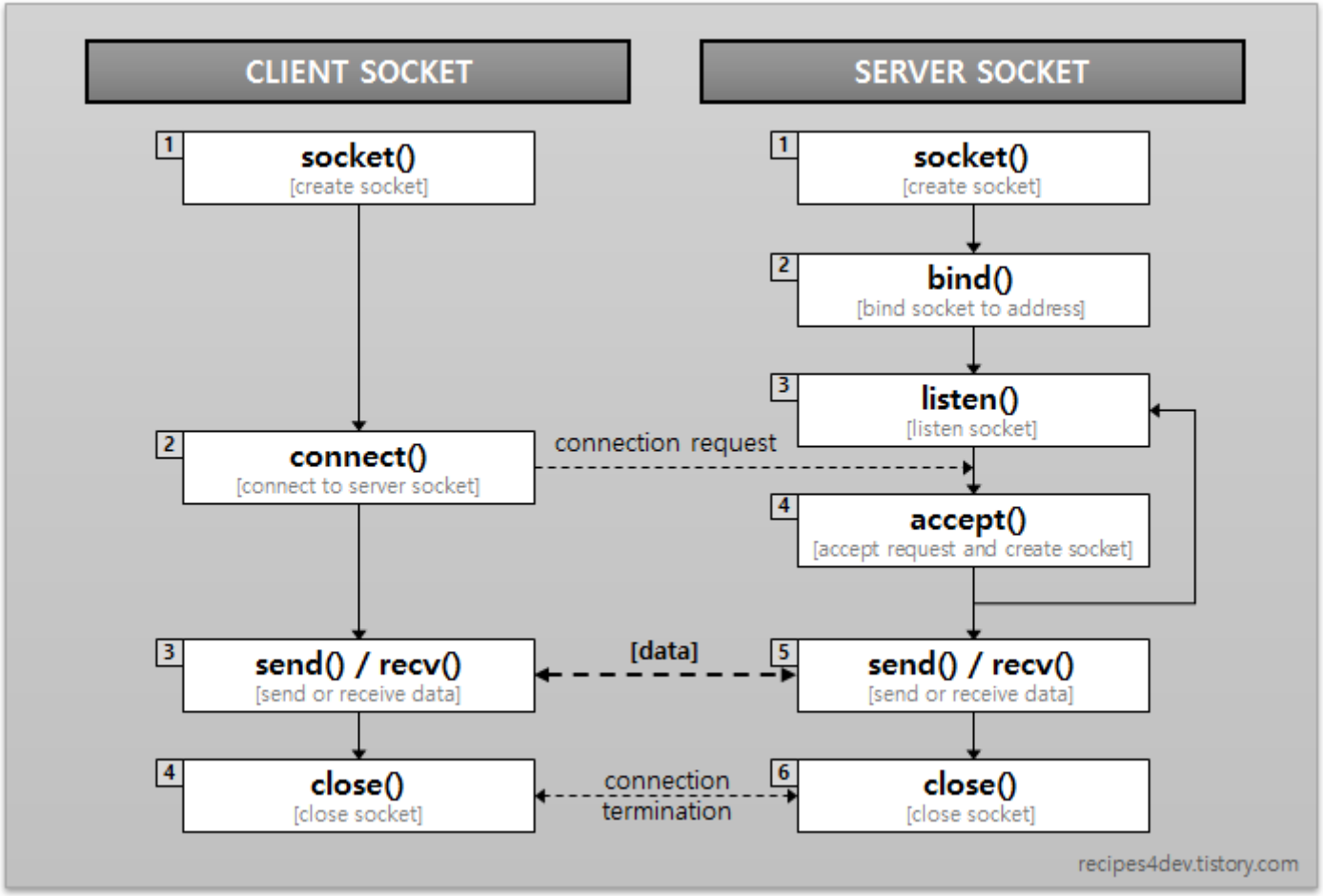
Socket Programming

소켓 프로그래밍

- code/socket에 주어진 소켓프로그래밍 예제를 확인해보자.
- Socket
  - 프로세스와 실제 네트워크를 연결해주는 **통로!**
  - 네트워크 프로그래밍 인터페이스(UC Berkeley에서 만든 Unix 표준 통신 인터페이스)
- 대부분의 웹 어플리케이션들이 Socket 인터페이스를 기반으로 만들어져있음

Socket Programming

소켓 프로그래밍



Pcap programming

Pcap 프로그래밍

- code/pcap에 주어진 pcap 프로그래밍 예제를 확인해보자.
- Pcap programming
  - libpcap을 다운 받아야 사용 가능하다.
    - sudo apt install libpcap-dev
  - 소켓과는 다르게 tcpdump(https://www.tcpdump.org/)에서 만든 **패킷 캡쳐** 라이브러리
  - 보통 네트워크 트래픽 관리를 할 때 이 라이브러리를 사용한다.
  - https://github.com/the-tcpdump-group/libpcap

Pcap programming

Pcap 프로그래밍

- https://www.tcpdump.org/pcap.html
- https://www.tcpdump.org/manpages/pcap.3pcap.html
- 위 사이트에서 여러 API들을 확인해볼 것

Description	Functions
pcap 핸들 열기	pcap_open, pcap_open_live, pcap_open_offline
pcap 핸들 닫기	pcap_close
packet 수신	pcap_next_ex
packet 송신	pcap_sendpacket

과제

과제설명

1. 소켓 프로그래밍

- code/socket 에 주어진 코드를 바탕으로 함.
- Client가 보낸 메시지를 relay하고 출력하는 server.cpp, client.cpp 작성

- ./server <port> [-e]
  - 옵션이 없을 경우에는 단순히 client가 보낸 메시지를 ./server에서 출력
  - -e 옵션이 주어지면, client가 보낸 메시지를 다시 client에게 전송
- ./client <IP address> <port>
  - 화면에 입력으로 서버로 메시지를 전송
  - 서버로 부터 오는 메시지를 recv해서 화면에 출력
    - 쓰레딩을 통해 구현하면 되지 않을까요?

과제

과제설명

2. 패킷 캡쳐 프로그램

- code/pcap 에 주어진 코드를 바탕으로 함.
- 패킷 캡쳐 시 Ethernet – IP – TCP 로 구성된 패킷에 대해서만 , source MAC, IP, PORT 와 Destination MAC, IP, PORT 정보를 출력하는 패킷 캡쳐 프로그램 작성
- 헤더를 인터넷에서 잘 찾아서, 변형해서 사용하거나! 직접 만들어 사용할 것!
  - https://github.com/afabbro/netinet/blob/master/ip.h
  - https://github.com/afabbro/netinet/blob/master/tcp.h
  - https://sites.uclouvain.be/SystInfo/usr/include/net/ethernet.h.html

과제

과제설명

- 제출 : ugonfor@gmail.com
- 메일 제목: [Network][디미고]본인이름
- 내용: 본인의 repository url을 포함하여!
- 제출기한 :
- 제출: ugonfor@gmail.com

- 보강 날짜 …

reference

- Computer Networking: A Top-Down Approach – 네트워크의 기본서
- https://gitlab.com/gilgil/sns/-/wikis/home
- 각종 위키피디아
- libpcap