

On Computer Networking

DK Moon

(dkmoon@mju.ac.kr)

쉬어 가는 코너

Q. 이 친구의 이름은?

A. Pig-let



쉬어 가는 코너

Q. 이것의 이름은?

A. Leaf-let



쉬어 가는 코너

Q. “물건 담긴 작은 봉투”를 영어로 하면?

packet | 'pakət |

noun

- 1 a paper or cardboard container, typically one in which goods are packed to be sold: *sow seeds 2 to 3 inches apart or as recommended on the seed packets.*
 - the contents of a packet: *he smoked **a packet of** cigarettes a day.*



- New Oxford American Dictionary

쉬어 가는 코너

- Kubernetes 는 이름을 줄여서 K8S 이라고도 부르고,
- Kube 라는 애칭으로도 부른다.
- Kube 에서 각 node 에서 돌고 있는 agent 이름은 Kubelet 이다.

취어 가는 코너

- Snip 은 가위로 잘라낸 것을 말한다.
- -et 를 붙인 형태인 snippet 은 (잘라낸 것 같은) 짧은 코드를 뜻한다.

JARGON 에 대해서...

- 특정 영역의 전문 용어를 지칭
- 그러나 jargon 자체도...
명확한 의미 전달을 위해
기존에 존재하던 단어에 기반해서 만든 것
 - 예시: Bandwidth = Band (대역) + Width(폭)
- 영어를 모국어로 쓰는 경우 어원을 유추하기 쉬우나
그렇지 않은 한국인은 단어일 뿐인 것을 너무 어렵게 생각함

네트워크란?

network | 'net,wərk |

noun

- ➡ 상호 연결되어있는 사람들이나 사물들로 이루어진 그룹이나 시스템
 - ➡ 사람/사물을 '점'으로 표시하고 그들간의 어떤 관계를 '선'으로 표시한 것
- *New Oxford American Dictionary*

네트워크 예시

점	선	네트워크 이름
뉴런	시냅스	신경망 (neural network)
사람	친분관계	사회 관계망 or 인맥 (social network)
기차역	선로	철도망 (rail transport network)
지역, 도시	도로	도로망 (road network)
전력 설비 (발전소, 변전소)	전력선	전력망 (power grid)
저항, 캐패시터, ...	전선	회로망 (electrical network)

네트워크 예시

점	선	네트워크 이름
뉴런	시냅스	신경망 (neural network)
사람	친분관계	사회 관계망 or 인맥 (social network)
기차역	선로	철도망 (rail transport network)
지역, 도시	도로	도로망 (road network)
전력 설비 (발전소, 변전소)	전력선	전력망 (power grid)
저항, 캐패시터, ...	전선	회로망 (electrical network)
컴퓨터	랜선	컴퓨터망 (computer network)


정리: Network

- Network: 개체(‘점’) 간의 관계 (‘선’) 을 표시한 것
- Networking: 개체(‘점’) 간의 관계를 맺게 하는 것

영어의 inter-

[←](#) [→](#) [↻](#) [🏠](#) [🔒 https://ko.wiktionary.org/wiki/inter-](#)

★ Bookmarks [📁 Programming](#) [📁 Linux](#) [📁 Xen](#) [📁 Market](#) [📁 Market Research](#) [📁](#)



위키낱말사전
말과 글의 누리

[대문](#)
[자유게시판](#)
[질문방](#)
[최근 바뀜](#)
[임의 문서로](#)
[편집실](#)
[새 낱말 쓰기](#)

[문서](#) [토론](#)

inter-

[목차 \[보이기\]](#)

영어 [\[편집 \]](#)

접두사 [\[편집 \]](#)

- 1. 몇몇 낱말의 앞에서 붙여서 그 '사이'라는 뜻을 더해 주는 접두사..

영어의 inter-

- inter-view
- inter-national
- inter-continental ballistic missile (ICBM)
- inter-stellar
- inter-currency
- ...



Inter-net(work) 라는 단어의 뜻

inter- : 사이
+ *network* : 점들을 선으로 연결한 덩어리

= (점들을 선으로 연결한) 덩어리 간의 연결

Network A



Network B



Inter-network



인터넷의 올바른 영어 표기

- internet(work) 은 일반적인 개념에 가까움

Q. 그렇다면 우리가 쓰는 “인터넷”의 올바른 영어 표기는?

A. the Internet

IP (Internet Protocol)

Q. 이제 “IP” 는 무슨 뜻일까요?

A. 네트워크를 연결하는 프로토콜

참고:

- 프로토콜 = “약속”

즉, 의사 소통을 위해 맞춰야 하는 것
다시 말해 “언어”

The Internet 의 목표

2. Fundamental Goal

The top level goal for the DARPA Internet Architecture was to develop an effective technique for multiplexed utilization of existing interconnected networks. Some elaboration is appropriate to make clear the meaning of that goal.

The components of the Internet were networks, which were to be interconnected to provide some larger service. The original goal was to connect together the original ARPANET⁸ with the ARPA packet radio network^{9,10}, in order to give users on the packet radio network access to the large service machines on the ARPANET. At the time it was assumed that there would be other sorts of networks to interconnect, although the local area network had not yet emerged.

ARPANET 이라는 네트워크와
ARPA 라는 패킷 라디오 네트워크의 연결



The Internet 의 목표

3. Second Level Goals

The top level goal stated in the previous section contains the word "effective," without offering any definition of what an effective interconnection must achieve. The following list summarizes a more detailed set of goals which were established for the Internet architecture.

1. Internet communication must continue despite loss of networks or gateways.
2. The Internet must support multiple types of communications service.
3. The Internet architecture must accommodate a variety of networks.
4. The Internet architecture must permit distributed management of its resources.
5. The Internet architecture must be cost effective.

1. 구성하는 네트워크 중 일부가 동작하지 않아도 계속 작동해야함
2. 다양한 통신 서비스 지원
3. 다양한 네트워크 수용 가능
4. 중앙집중식이 아닌 분산처리 방식의 자원 관리
5. 비용 효율적
6. 적은 비용으로 호스트 추가 가능
7. 누가 어느 정도 리소스를 쓰는지 추적 가능

The Internet 의 핵심 목표 정리

- 0. ARPANET 이라는 네트워크와 ARPA 패킷 라디오 네트워크 연결
- 1. 구성하는 네트워크 중 일부가 동작하지 않아도 계속 작동해야함
- 2. 다양한 통신 서비스 지원
- 3. 다양한 네트워크 수용 가능
- ...

The Internet 의 목표는 잘 달성되었을까?

- 0. ARPANET 이라는 네트워크와 ARPA 패킷 라디오 네트워크 연결
- 1. 구성하는 네트워크 중 일부가 동작하지 않아도 계속 작동해야함
- 2. 다양한 통신 서비스 지원
- 3. 다양한 네트워크 수용 가능
- ...

The Internet 의 목표는 잘 달성되었을까?

0. ARPANET 이라는 네트워크와 ARPA 패킷 라디오 네트워크 연결

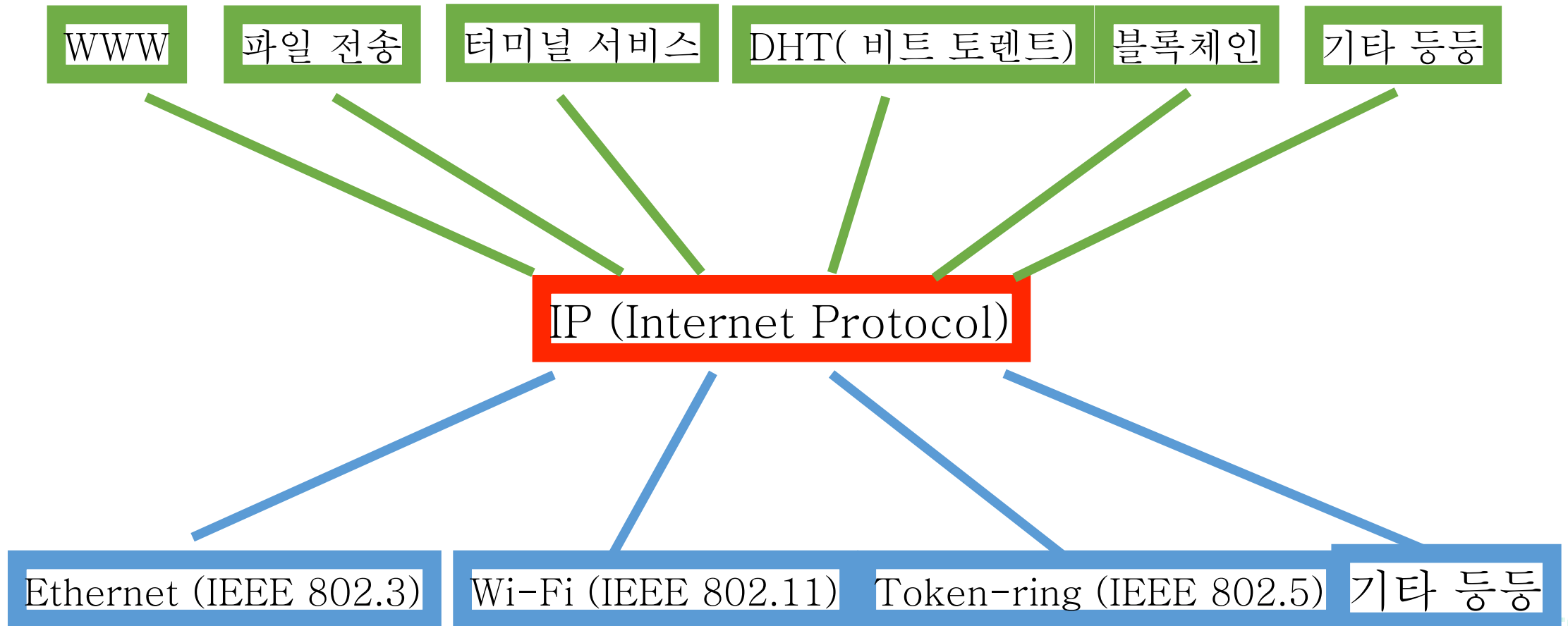
1. 구성하는 네트워크 중 일부가 동작하지 않아도 계속 작동해야함

2. 다양한 통신 서비스 지원

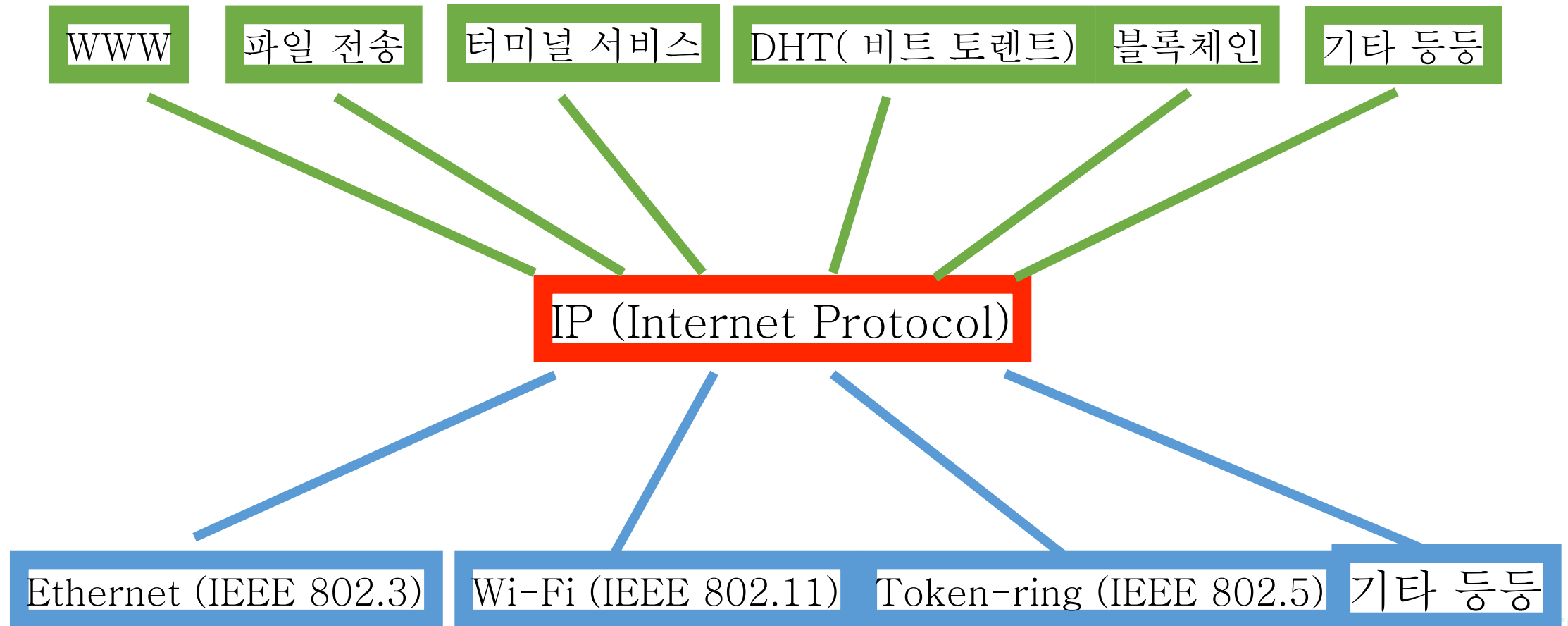
3. 다양한 네트워크 수용 가능

...

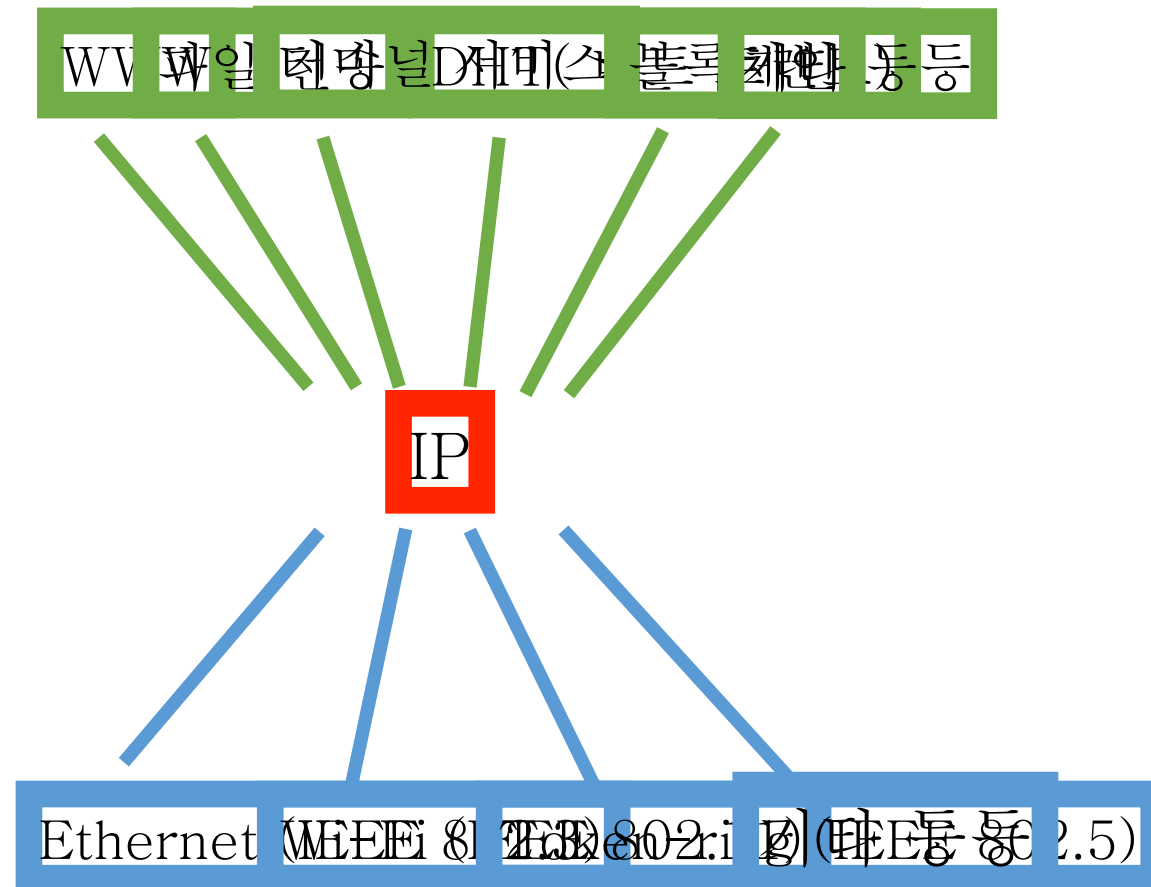
The Internet 의 목표는 잘 달성되었을까?



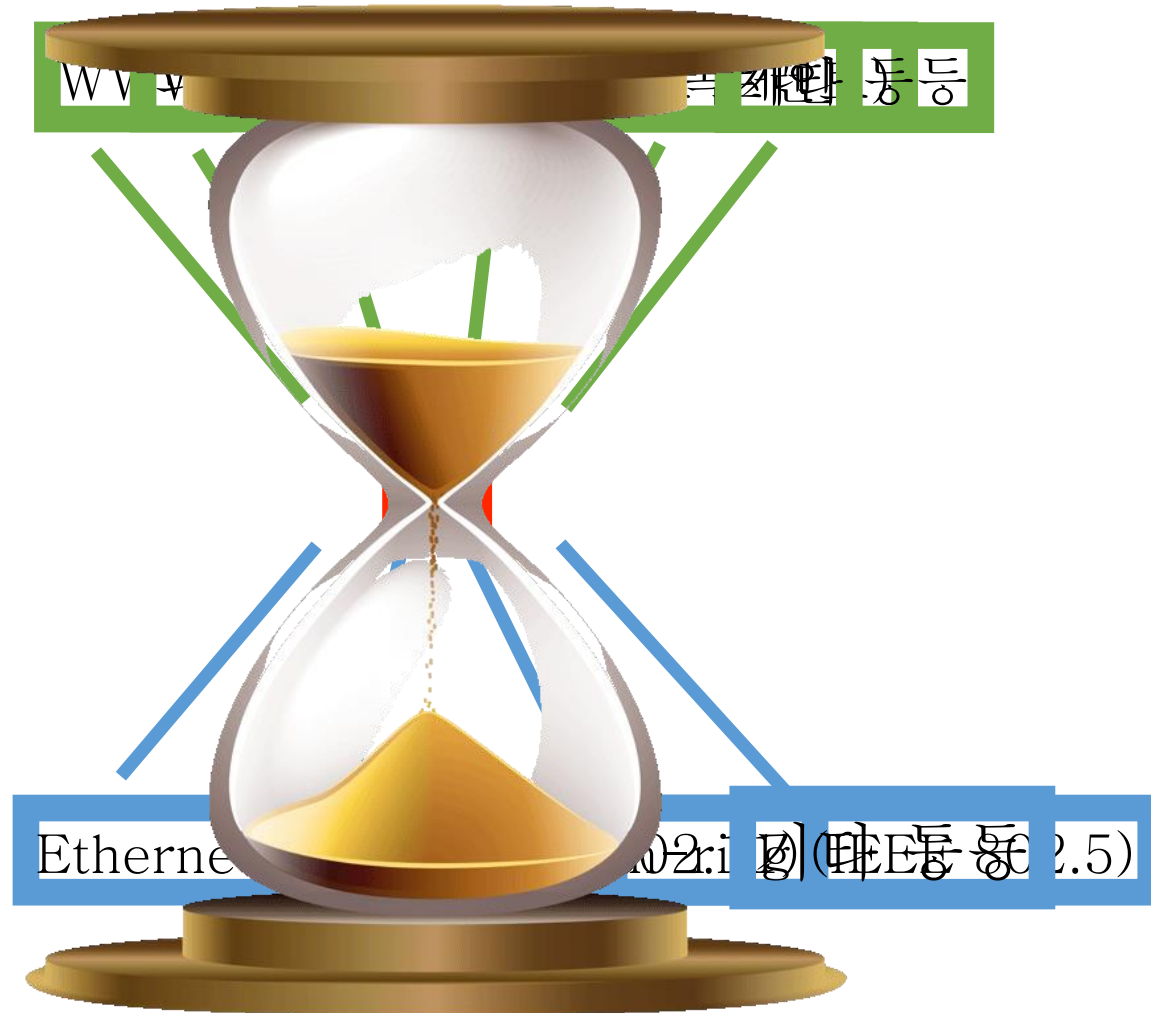
어떤 모양이 보이나요?



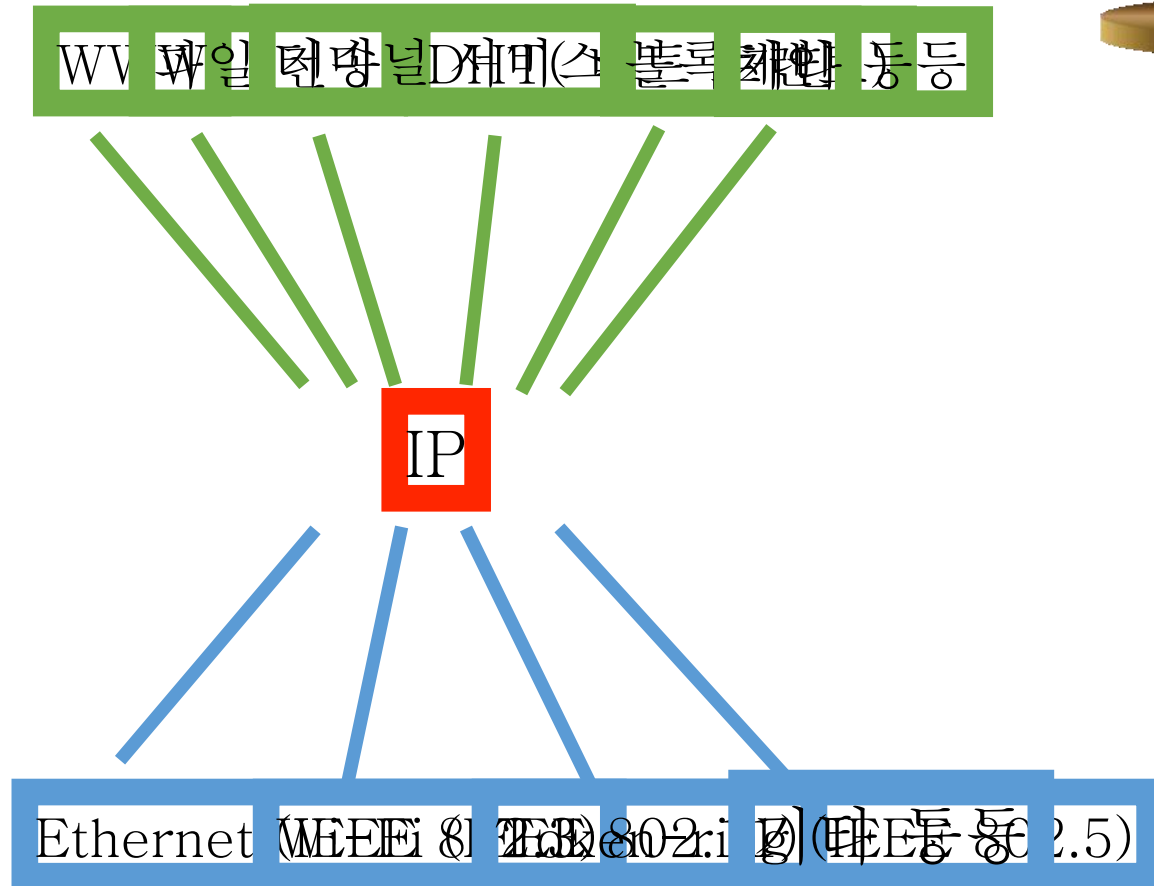
이러면 좀 더 잘 보이나요?



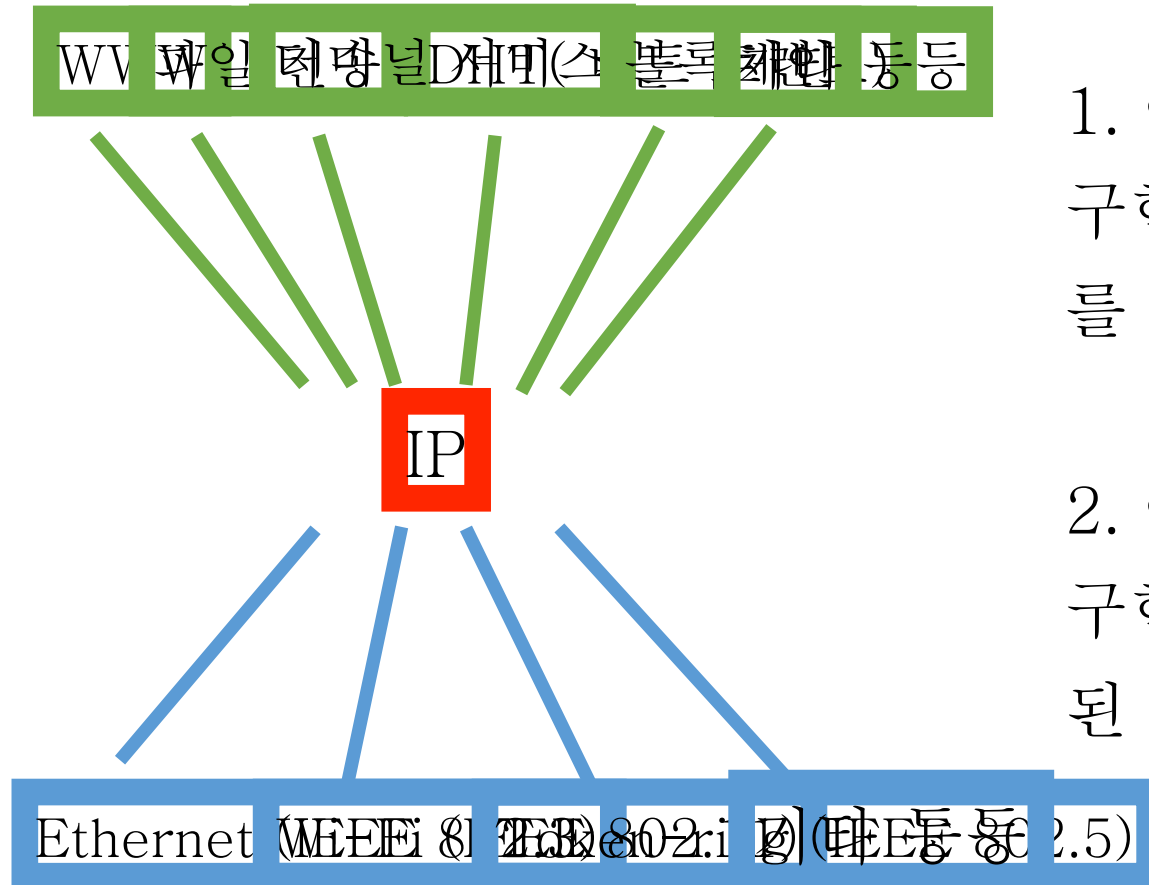
이러면 좀 더 잘 보이나요?



Hour glass 모델. IP 는 그 중 허리



Hour glass 모델의 허리라는 의미는?



1. 어떤 물리적 연결 기술이든 IP 만 구현하면 다양한 서비스(소프트웨어)를 돌릴 수 있다.

2. 어떤 서비스(소프트웨어)든 IP 로만 구현하면 다양한 물리적 연결 기술로 된 네트워크에서 동작한다.

컴퓨터공학에서의 계층화 (Layering)

모든게 들어가 있는
거대한 하나의 덩어리



저수준 기능과
이를 사용해 만들어지는
고차원의 기능으로 구분

컴퓨터공학에서의 계층화 (Layering)



계층화(Layering)의 장점

- 단순화
 - 각 계층은 자기가 제공할 기능만 생각하면 된다. (functionality)
 - 각 계층은 자기 바로 아래 계층을 어떻게 쓰는지만 알면 된다. (interface)
- 문제 해결의 편의성
 - 문제가 있는 계층만 디버깅하면 된다.
 - 각 계층이 단순하기 때문에 문제를 해결하기도 쉽다.
- 진화의 편의성
 - 각 계층은 바로 위 계층에 알려준 “어떻게 쓰는지”만 유지하면 된다.
 - 그 안에서 자유롭게 기능을 개선/추가 할 수 있다.
 - 컴퓨터 공학의 격언 “계층 추가하면 뭐든 쉽게 해결 가능”

계층화(Layering)의 단점

- 잠재적 비효율성
 - 각 계층을 넘나드는 것이 비효율적일 수 있다.
 - 바로 아래 계층이 아니라 아래아래 계층처럼 계층을 건너뛰어야 될 때 괴롭다.
 - 어떤 계층을 건드리면 그 것을 사용하는 그 위의 모든 계층이 영향을 받는다.

계층화(Layering)에서의 황금률

하위 계층에 뭘 넣지 마라.

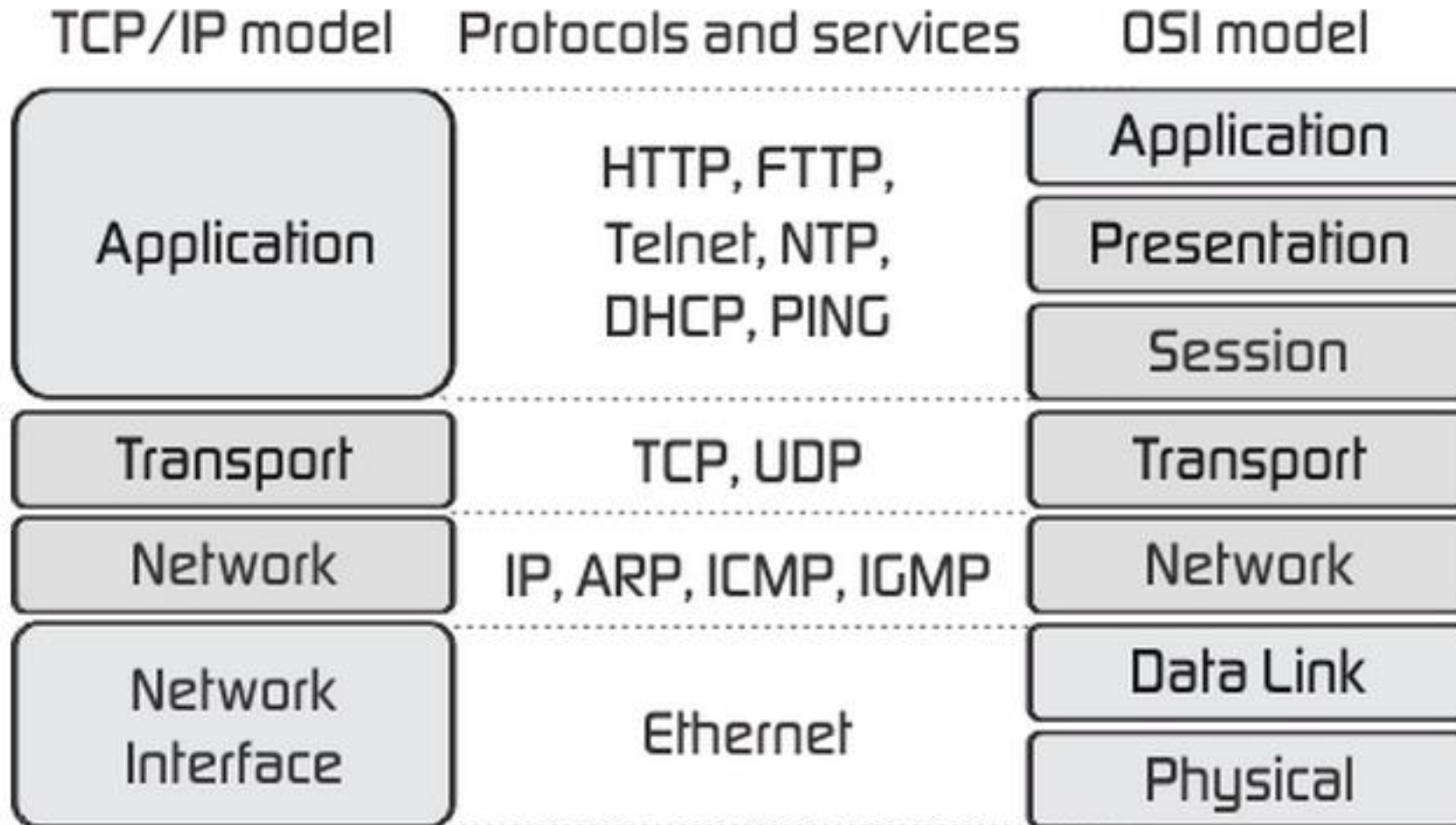
그게 성능상 절대적으로 필요한 것이 아니라면...

OSI 7 계층 모형 (OSI Reference Model)

- 영어 이름에서 기준 모델 (reference model) 이라고 한 것에 주목.
- 즉, 실제 구현으로 존재하는 것이 아님
- 대신 우리가 계층 구조를 만들 때 기준이 됨을 의미
- 간혹 OSI7 계층 모델을
“아무도 안 쓰는 무의미 한 것” 으로 폄하하는
사람들이 있는데, 잘못 이해한 것임.



TCP/IP 에서의 OSI Model 적용



계층 구조에서 데이터 처리 (보내는 쪽)



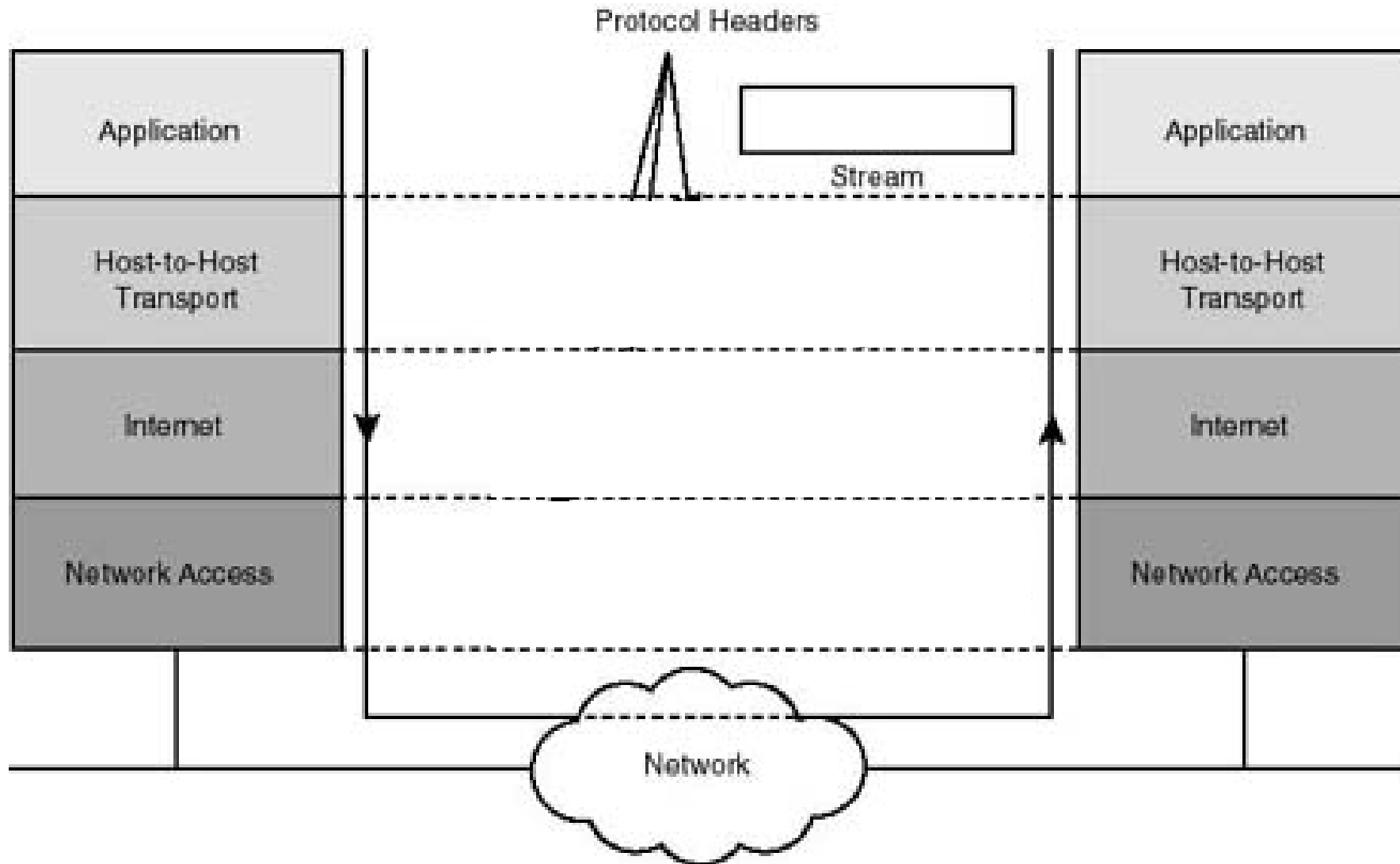
계층 구조에서 데이터 처리 (받는 쪽)



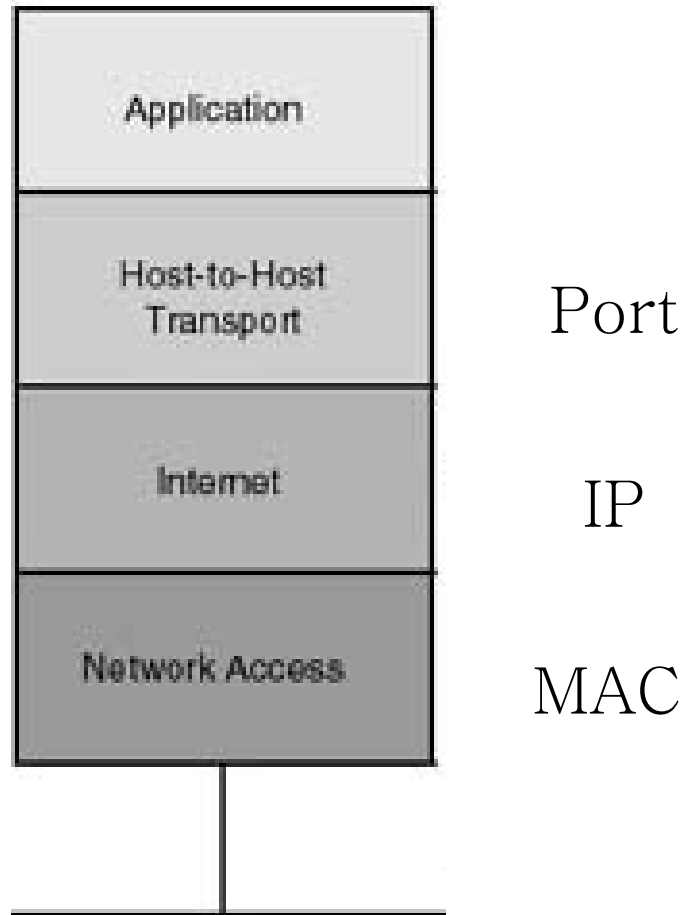
정리: 계층 구조에서 데이터 처리

- (보내는 쪽) 보내는 사람, 받는 사람을 기재한 더 큰 박스에 넣는다.
- (전송) 최종의 박스 (제일 큰 거) 를 보낸다.
이 때 박스의 받는 사람 정보를 참조해서 전송한다.
- (받는 쪽) 바깥쪽 박스를 제거하고
작은 박스를 꺼내 위 계층에 전달한다.

컴퓨터 네트워크 관점에서 박스 안에 박스 넣기



컴퓨터 네트워크 각 계층의 주소 요소



다시 쉬어 가는 코너

Q. 항구를 영어로 하면?

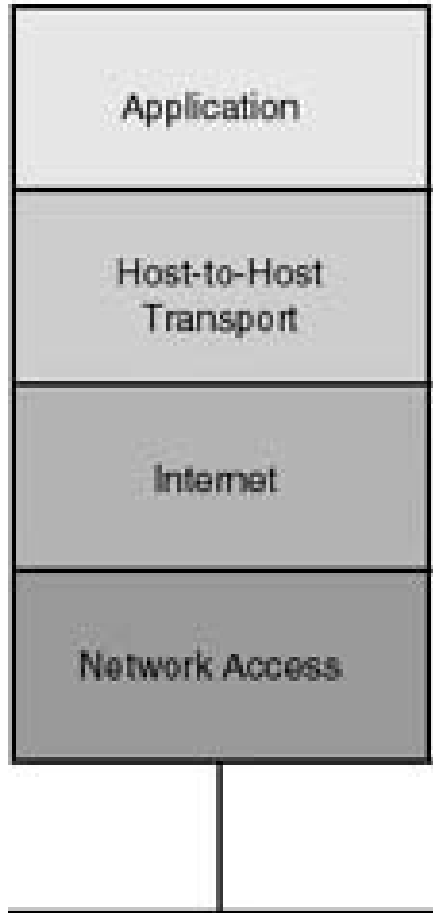
A. Port

다시 쉬어 가는 코너

Q. 공항을 영어로 하면?

A. Airport

컴퓨터 네트워크 각 계층의 주소 요소



Port: “IP” 라는 국가에 왔는데 어느 항구로 갈까요?

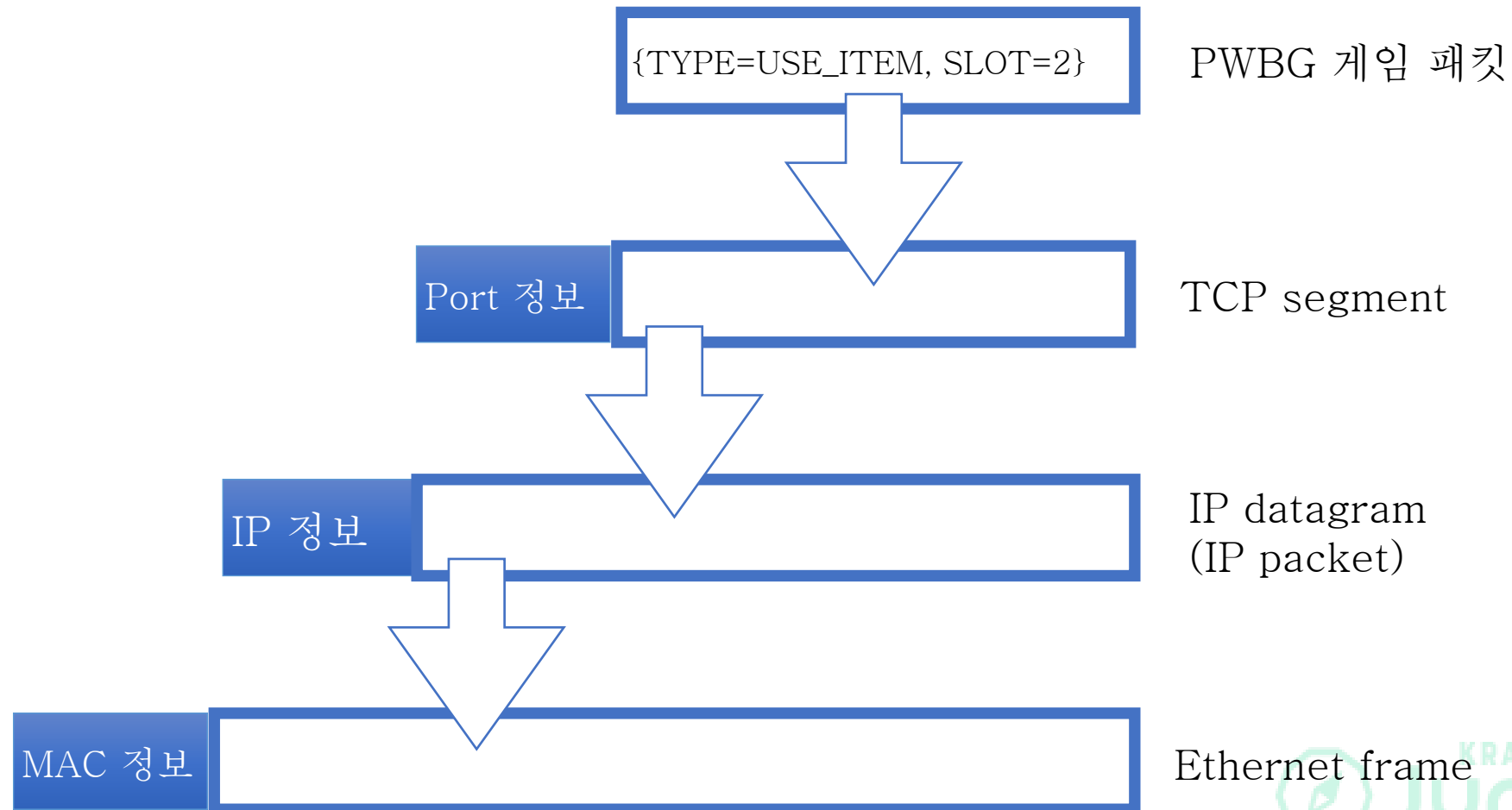
IP

MAC

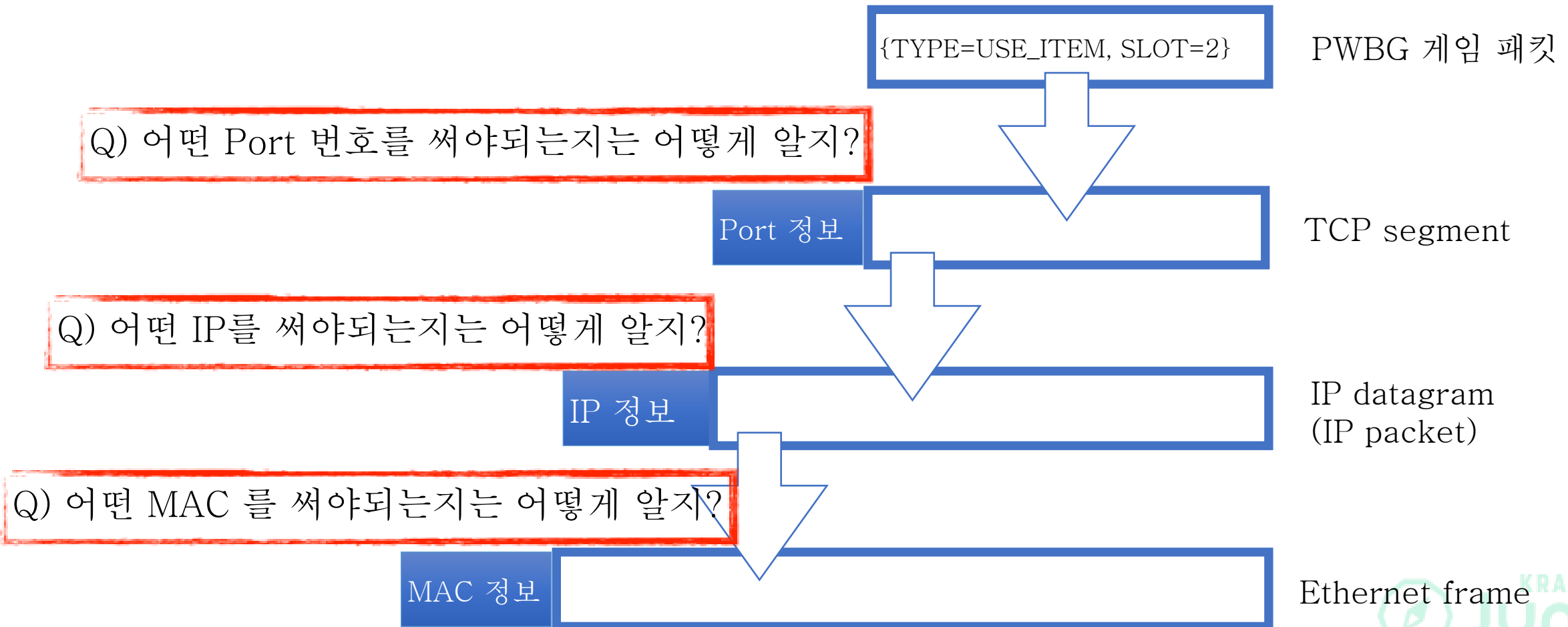


계층 구조에서 데이터 처리 TCP/IP 예시

“Player Well-known’s Battleground” 에서 2번 슬롯 아이템 사용
하기

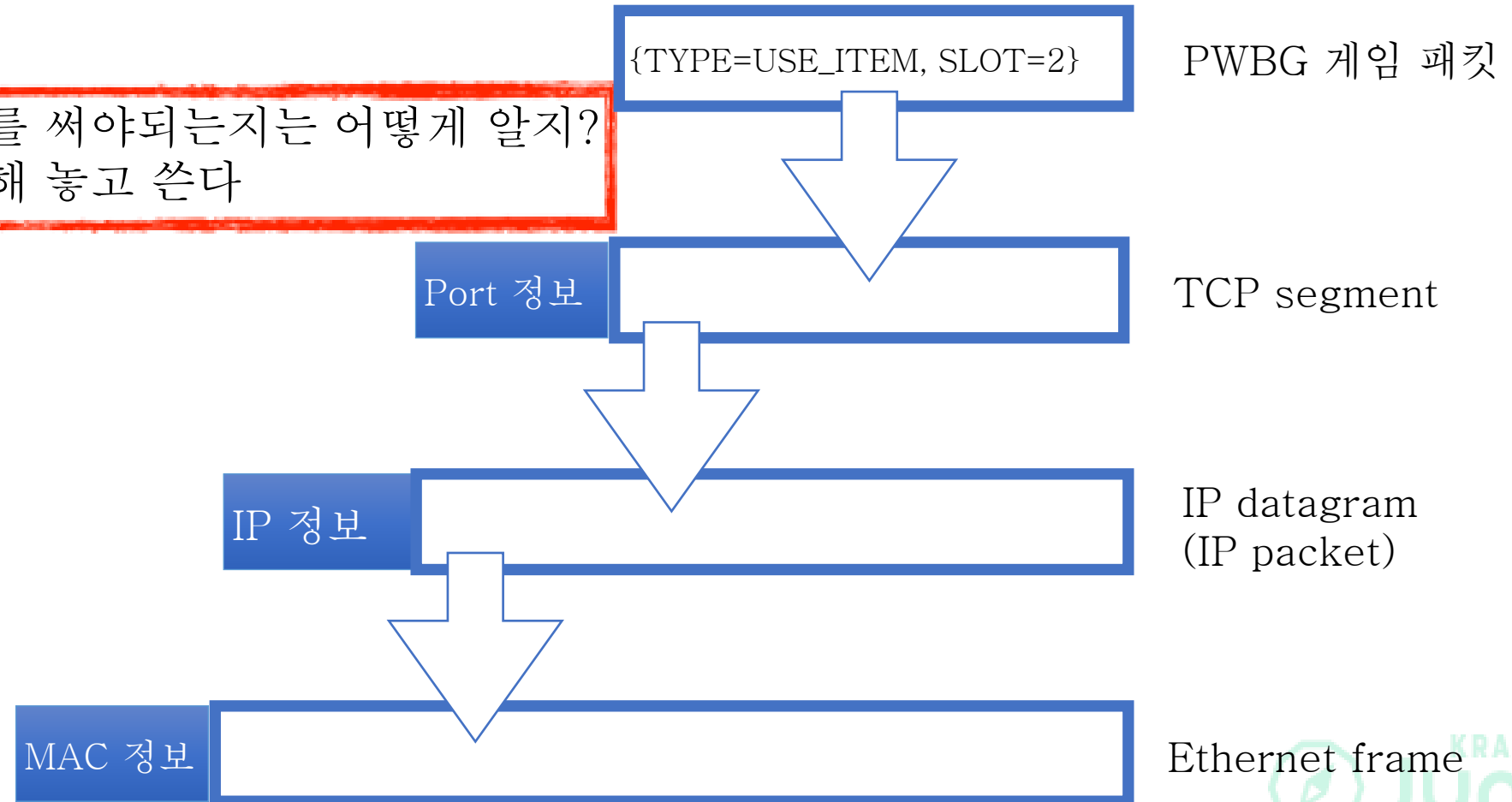


어떻게 주소 요소들을 알아낼까?

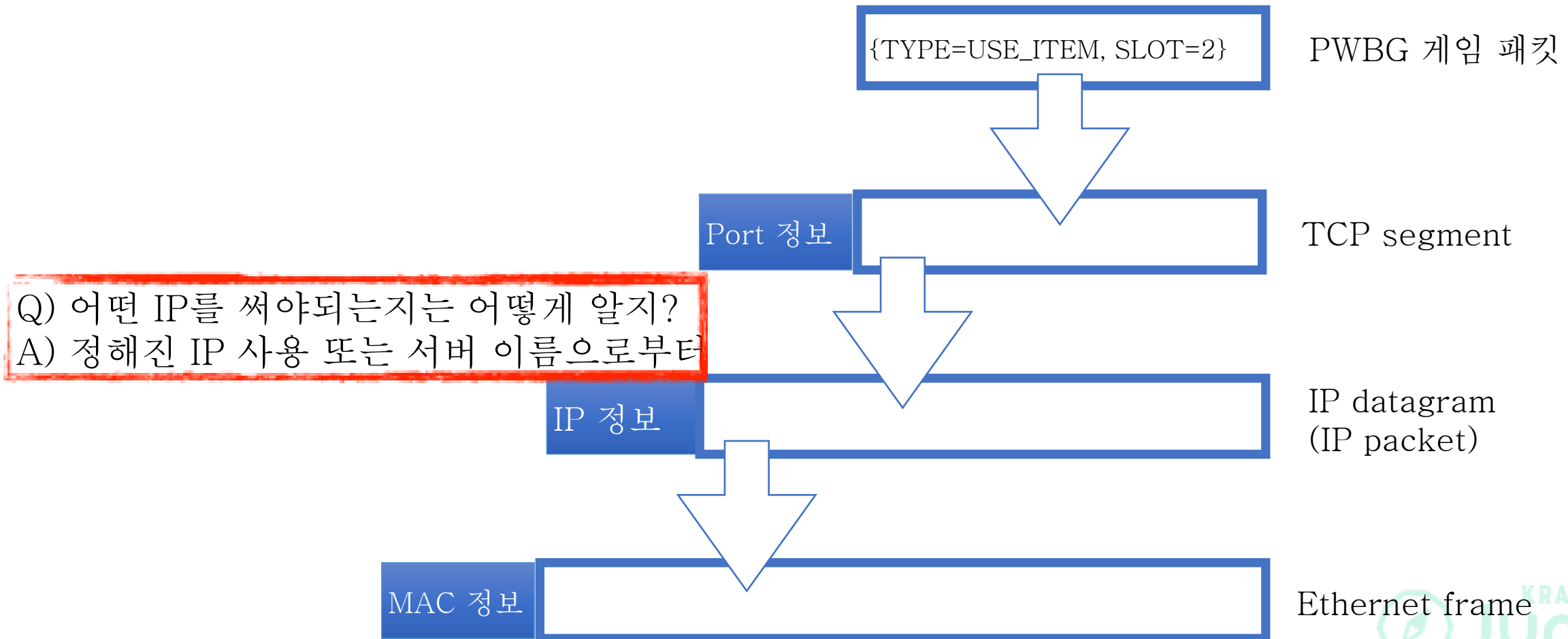


어떻게 주소 요소들을 알아낼까?

Q) 어떤 Port 번호를 써야되는지는 어떻게 알지?
A) Port 번호는 정해 놓고 쓴다



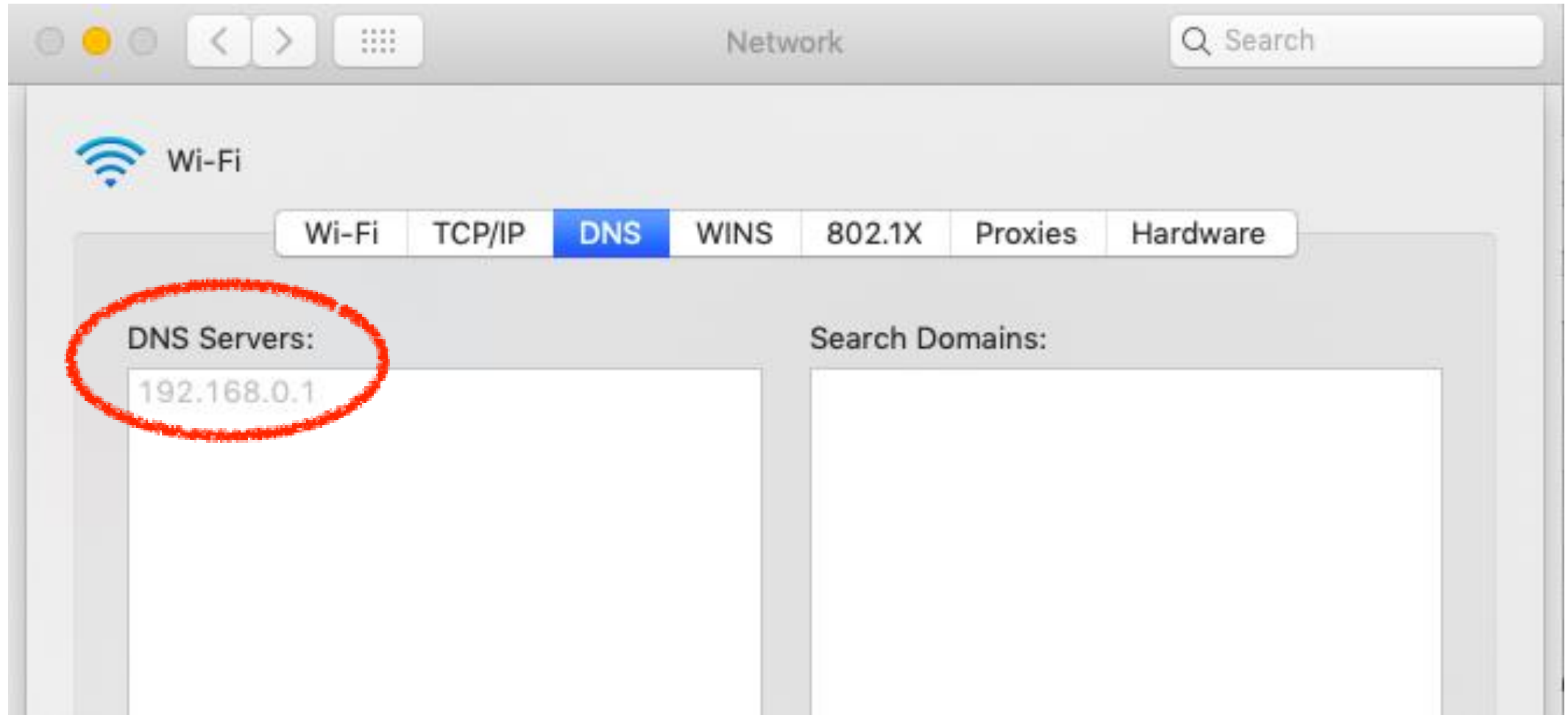
어떻게 주소 요소들을 알아낼까?



서버 이름에서 IP 알아내기

- 사람에게서는 숫자 나열보다 이름이 기억하기 더 쉽다.
- 하지만 the Internet 은 IP 로 통신한다.
- 그렇다면 이름과 IP 주소를 매칭해서 기억하고 있으면 어떨까?
그리고 필요할 때 주어진 이름에서 IP 를 찾아서 알려주는 것이지.
- DNS: 그것을 위한 서비스
- DNS 서버: 서버 이름과 IP 매칭을 기억하는 저장소
누군가 서버 이름가지고 IP 를 물어보면 대답해주는 역할
- DNS resolution: 서버 이름으로부터 IP 를 알아내는 행동

실생활에서 DNS



실생활에서 DNS

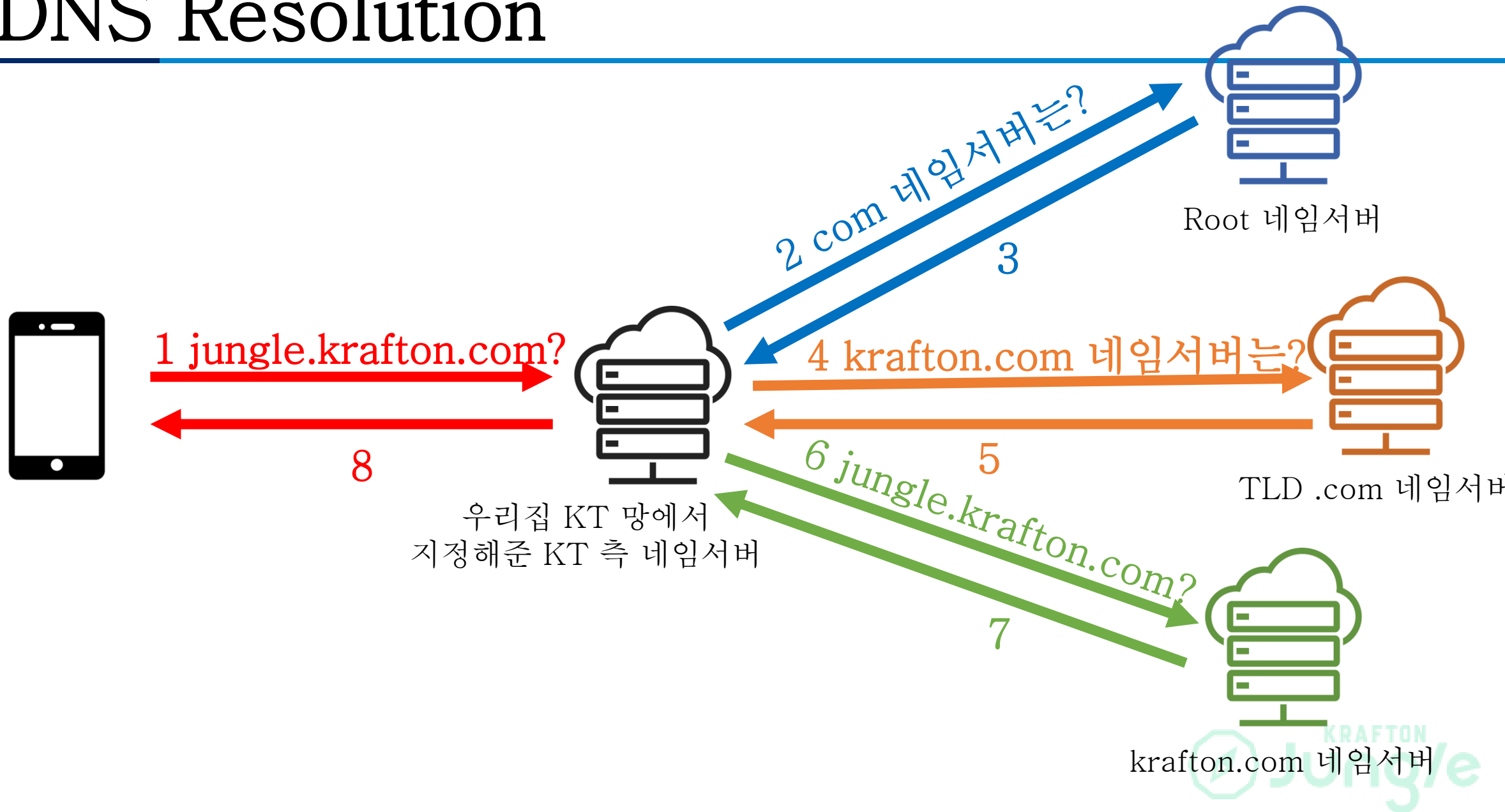
아시아-태평양 서울 리전(AP-NorthEast-2)의 Amazon EC2 DNS 확인(Resolution) 이슈 요약

Read in [English](#)

2018년 11월 22일 서울 리전(AP-NORTHEAST-2)에서 발생한 서비스 중단 상황에 대해 추가적으로 정보를 알려드립니다. 당일 한국시간 오전 8시 19분에서 9시 43분까지 서울 리전에서 EC2 인 이슈가 있었습니다. 이는 EC2 인스턴스에 재귀 DNS 서비스를 제공하는 EC2 DNS 확인 서버군(resolver fleet) 중 정상 호스트 수가 감소했기 때문입니다. 정상 상태의 호스트 수가 이전 수준으로 DNS 확인 서비스는 복원되었습니다. 이번 이슈에서 EC2 인스턴스의 네트워크 연결 및 EC2 외부의 DNS 확인 과정은 영향을 받지 않았습니다.

DNS 확인 문제의 근본 원인은 설정 업데이트 시 서울 리전의 EC2 DNS 확인 서버군의 최소 정상 호스트를 지정하는 설정을 잘못 제거한 것에 기인합니다. 이로 인해 최소한의 정상 호스트 구성 기본 것으로 해석되어, 정상 서비스 호스트 숫자가 줄어들었습니다. EC2 DNS 확인 서버군의 정상 호스트 용량이 감소함에 따라, 고객 EC2 인스턴스 내의 DNS 쿼리가 실패하기 시작했습니다. AWS 엔지니어는 오전 9시 21분에 서울 리전 내의 DNS 확인 문제가 통보되었고, 즉시 문제 해결에 나섰습니다. AWS는 먼저 더 이상 정상 호스트가 서비스에서 제거되는 것을 방지함으로써 추가적인 영향이 없음을 확인했으며, 복구 시간이 대부분이 이 작업에 사용되었습니다. 한국 시간 오전 9시 43분에 EC2 인스턴스의 DNS 질의 문제를 완전히

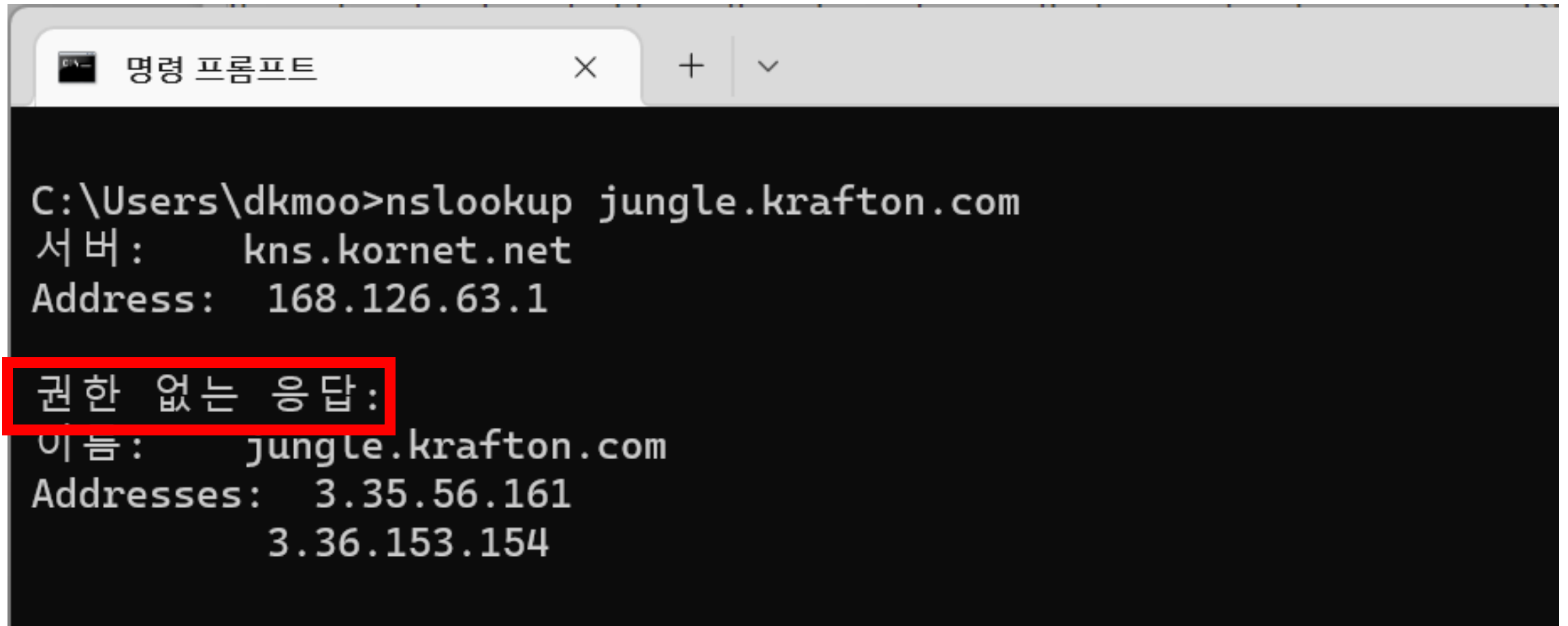
DNS Resolution



DNS Caching

- jungle.krafton.com 의 예에서 알 수 있듯
DNS resolution 은 iterative 한 작업이다.
 - root → com → krafton.com → jungle.krafton.com
- 문제점
 - Root 와 Top-level domain (TLD) 를 담당하는 서버는 엄청난 부하를 받는다.
 - Resolution 까지 많은 시간이 걸린다.
- DNS caching
 - DNS record 는 빈번히 바뀌는 정보가 아니다. → Caching 에 적합
 - Caching 을 통해 high scalability & low latency 둘 다 이룰 수 있다.
 - 예: root NS 캐싱, TLD NS 캐싱, 개별 도메인의 NS 캐싱, DNS 쿼리 결과 캐싱

DNS Caching 예시



A screenshot of a Windows Command Prompt window titled "명령 프롬프트". The window shows the output of the command `nslookup jungle.krafton.com`. The output includes the server used for the lookup (`kns.kornet.net`), its address (`168.126.63.1`), and the response for the domain `jungle.krafton.com`, which has two IP addresses: `3.35.56.161` and `3.36.153.154`. The text "권한 없는 응답:" (Unauthorized response:) is highlighted with a red rectangular box.

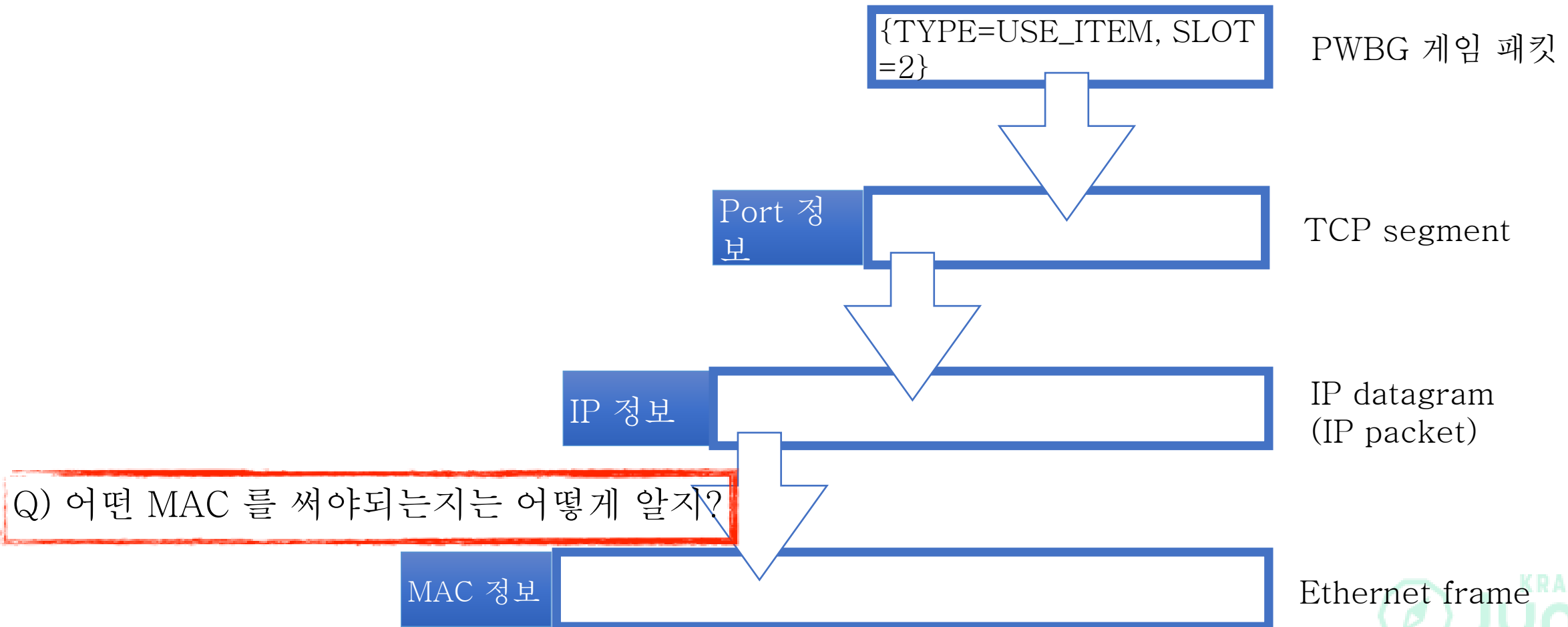
```
C:\Users\dkmoo>nslookup jungle.krafton.com
서버:      kns.kornet.net
Address:   168.126.63.1

권한 없는 응답:
이름:      jungle.krafton.com
Addresses: 3.35.56.161
           3.36.153.154
```

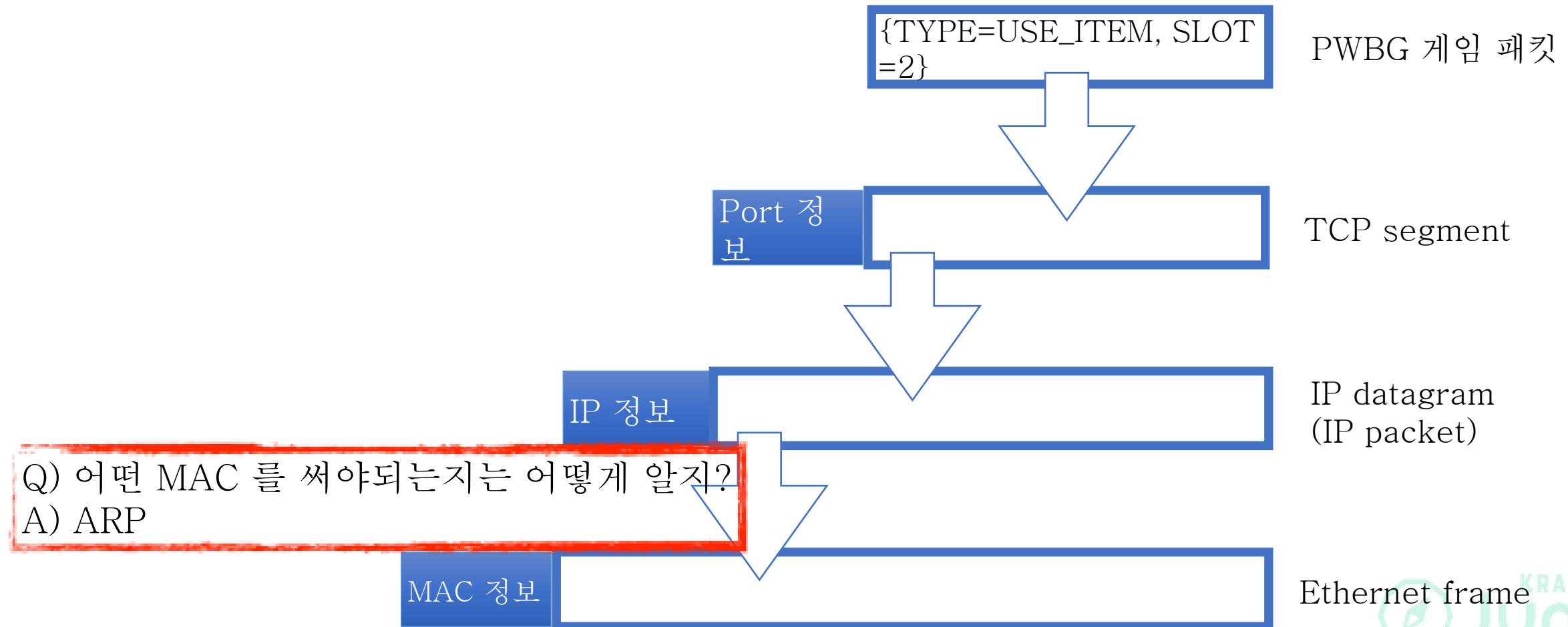
IP 에서 MAC 알아내기

- IP 는 네트워크를 연결한다.
- 그말인즉, IP 보다 아래에 있는 MAC 은 한 네트워크 안에서만 동작
- 그렇다면 MAC 필요할 때
네트워크에 “이 IP 주소 쓰는 사람은 MAC 알려줘” 라고 하면 어떨까?
딴 네트워크로 안 넘어가니까 그냥 전체 방송(broadcast) 해서...
- ARP: 이런 절차를 위한 프로토콜

어떻게 주소 요소들을 알아낼까?



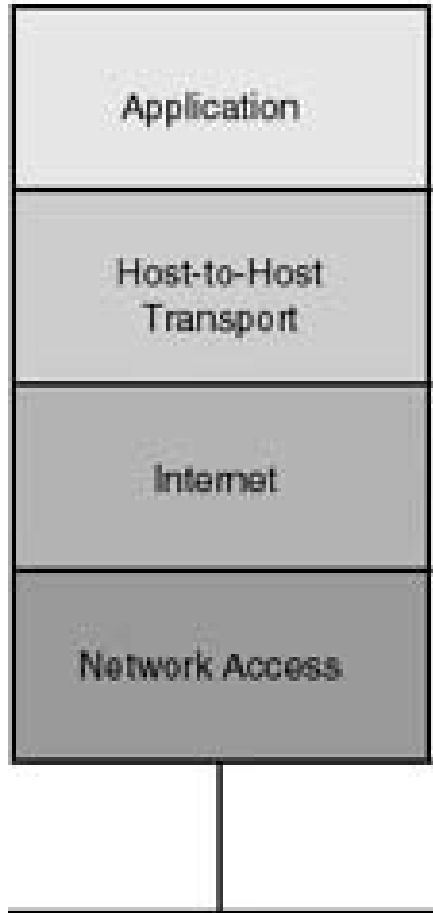
어떻게 주소 요소들을 알아낼까?



서버 관리자가 보는 ARP

```
dkmoon — ubuntu@ip-10-0-0-143: /var/www/coinonecore.co.kr/data — -bash — 80x24
exchange-commit3:~ dkmoon$ arp -a
? (192.168.0.1) at f4:f2:6d:fb:f5:2 on en0 ifscope [ethernet]
? (192.168.0.103) at f8:e9:4e:6c:25:c8 on en0 ifscope [ethernet]
? (192.168.0.111) at 28:cf:e9:17:78:c3 on en0 ifscope [ethernet]
? (224.0.0.251) at 1:0:5e:0:0:fb on en0 ifscope permanent [ethernet]
? (239.255.255.250) at 1:0:5e:7f:ff:fa on en0 ifscope permanent [ethernet]
exchange-commit3:~ dkmoon$
```


네트워크 연결 프로토콜인 IP



Q) 접속할 서버가 다른 네트워크에 있다면?

Port

IP

MAC



쉬어 가는 코너

- Gateway. 영어 단어입니다. 한글로 무슨 뜻일까요?

어학사전

gateway

영어사전 단어·숙어 1-5 / 506건

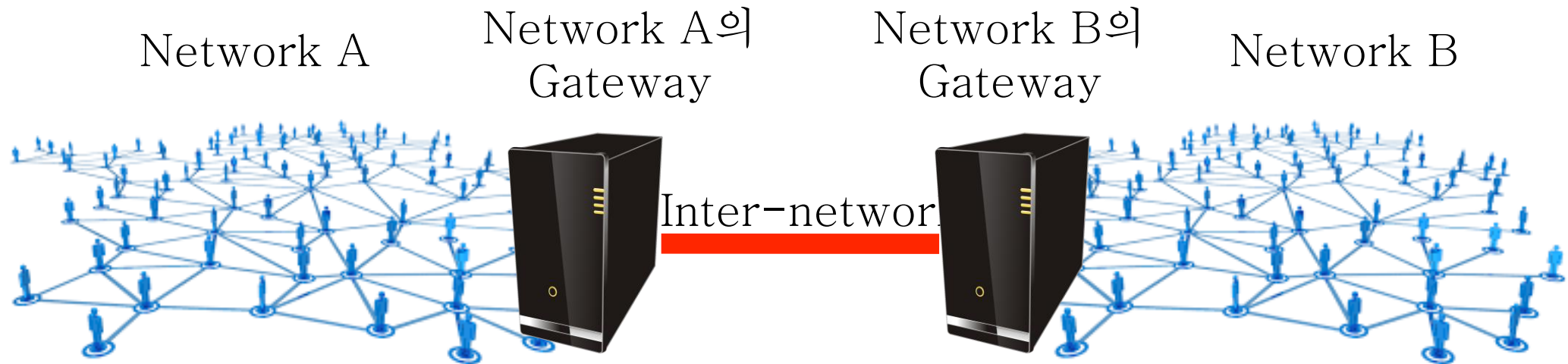
gateway 미국·영국 ['geɪtweɪ]  영국식  ★

1. (문이 달려 있는) 입구
2. (...로 가는) 관문
3. ~에 이르는 길

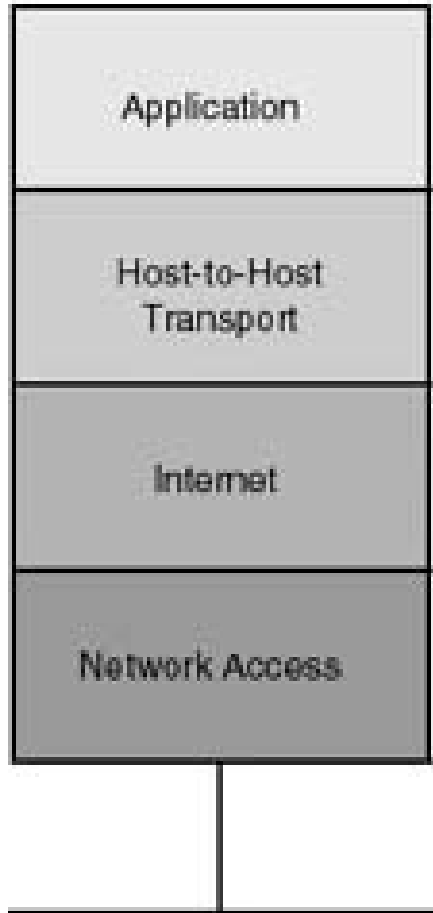
Gateway

- 입출입을 위한 관문
- 인천공항, 김포공항, 인천항, 부산항, ... → 대한민국의 gateway
- 네트워크 세상에서는 → 내 네트워크의 입출입을 위한 관문 서버

Network Gateway



네트워크 연결 프로토콜인 IP



Q) 접속할 서버가 다른 네트워크에 있다면?

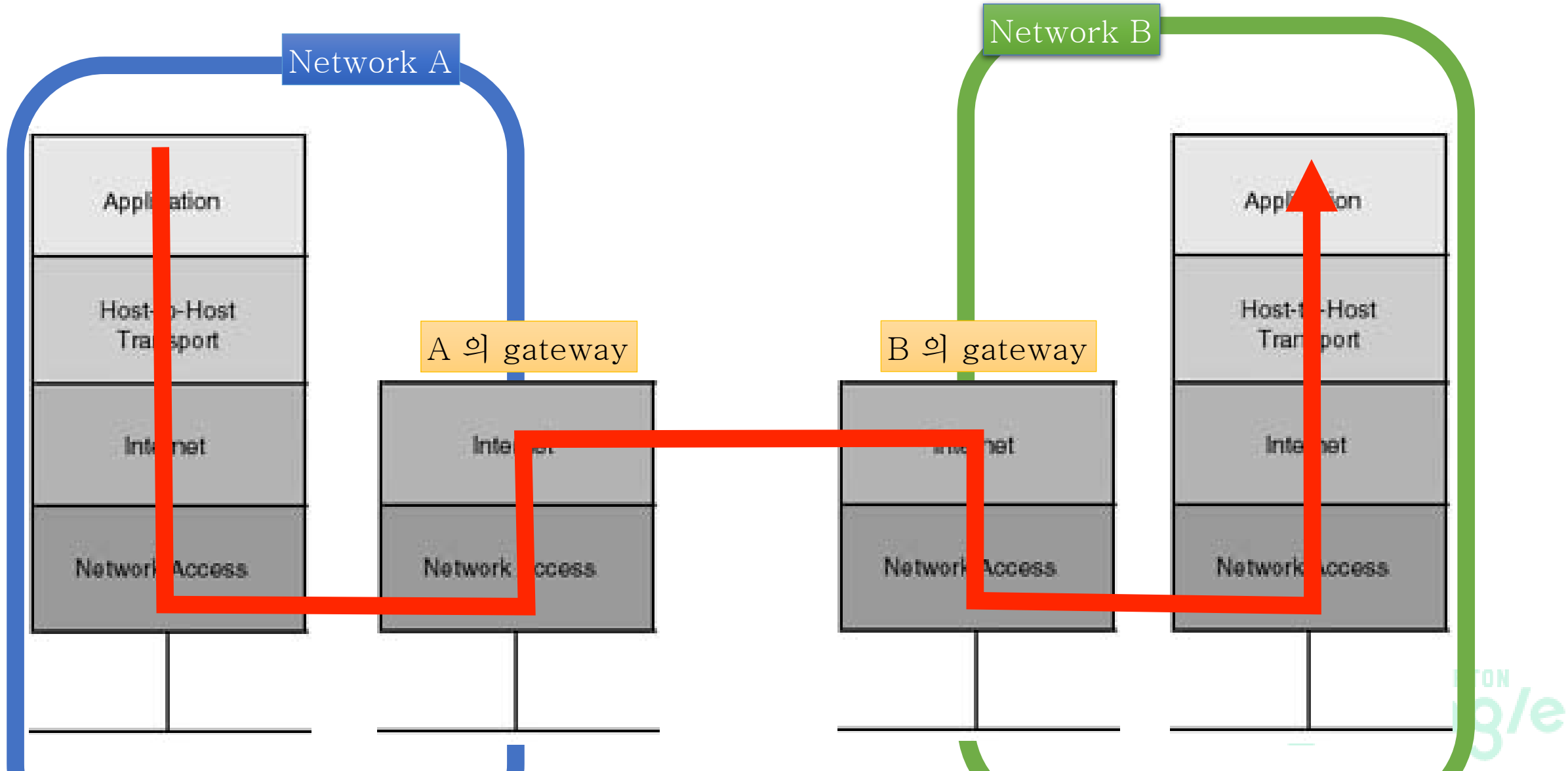
A) Gateway 에게 던져 놓고 맡긴다.

Port

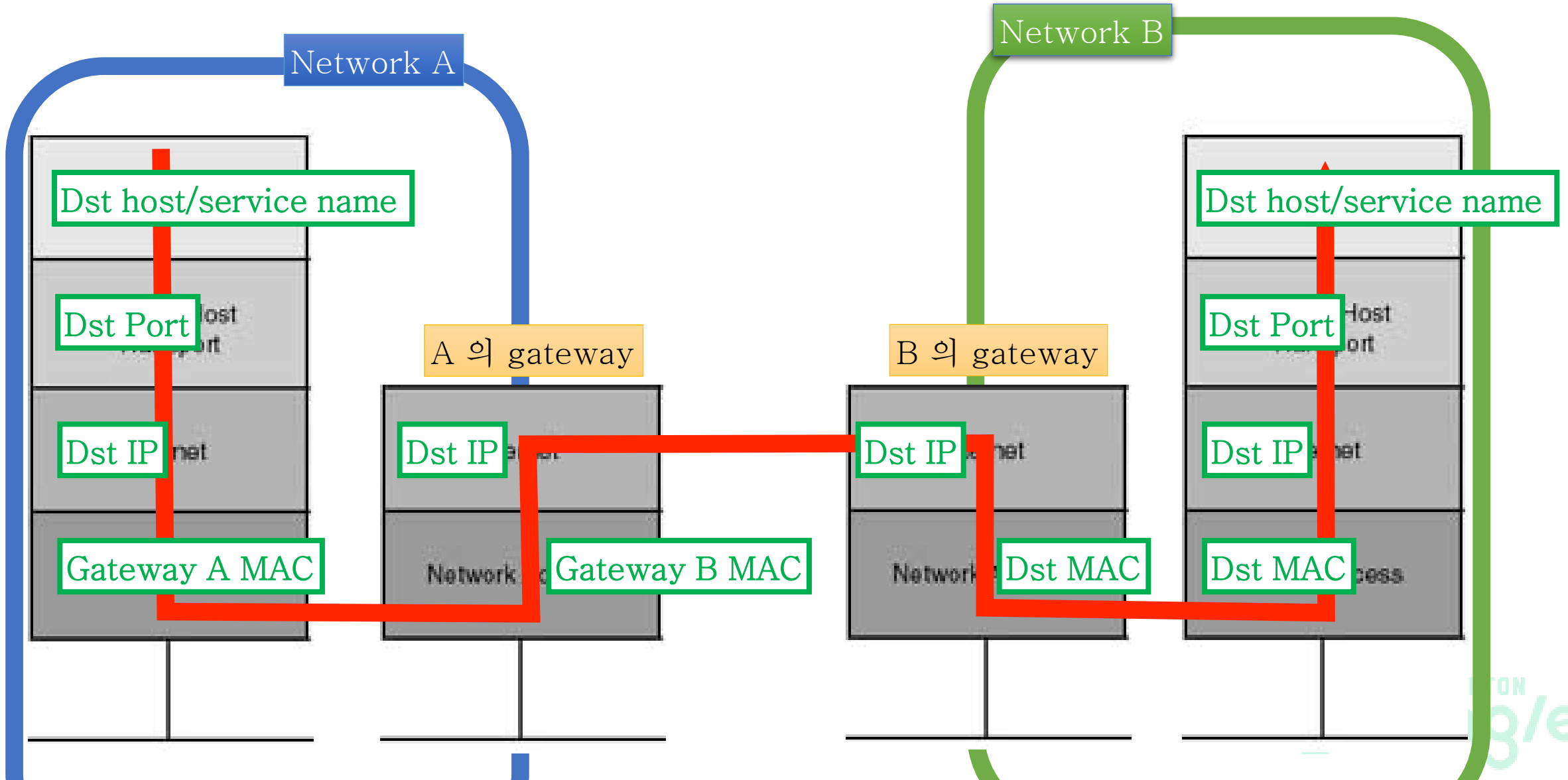
IP

MAC

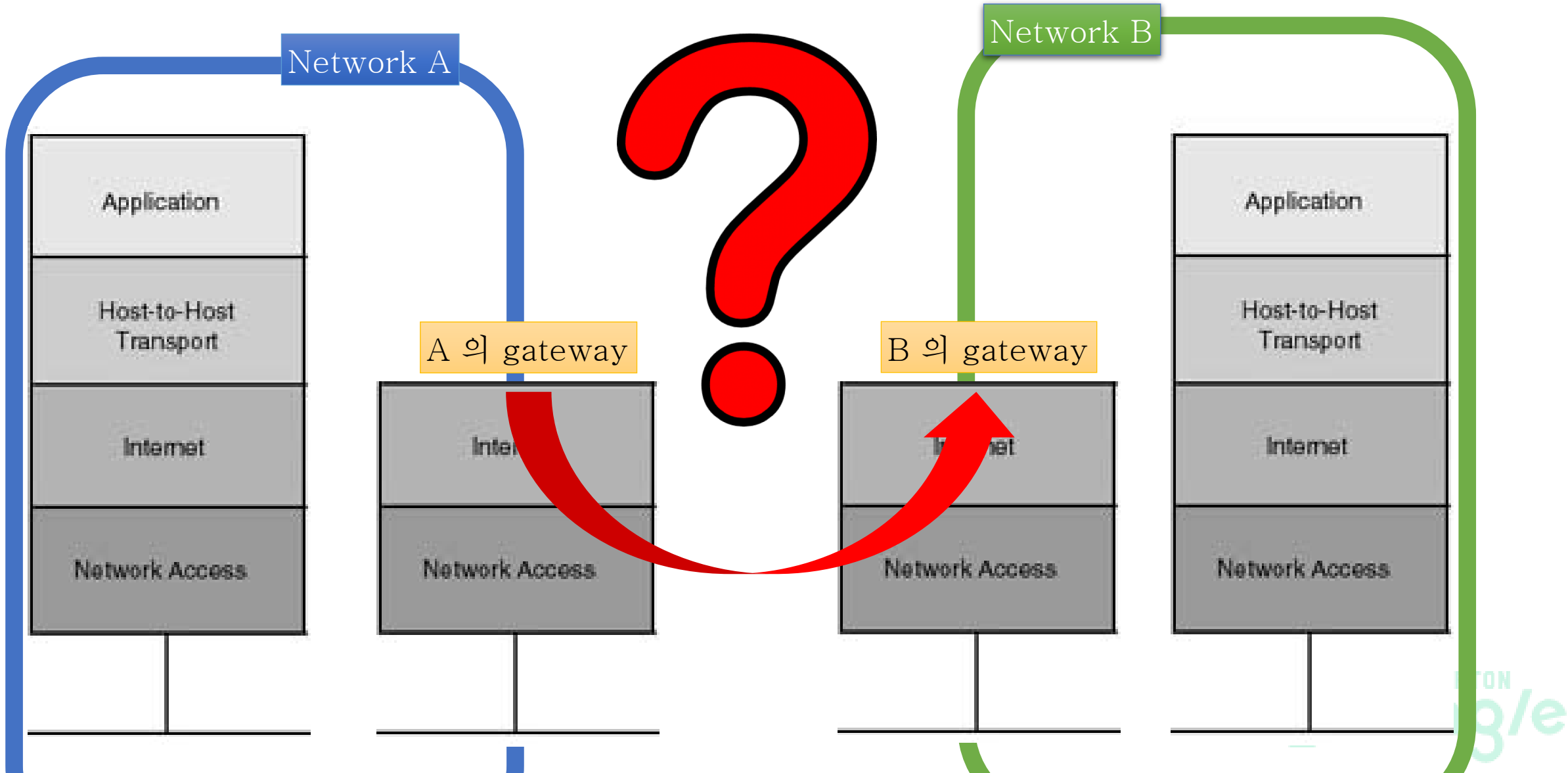
IP: Hop-by-Hop Network



심화문제: 트래픽에서 업데이트 되는 주소 요소?



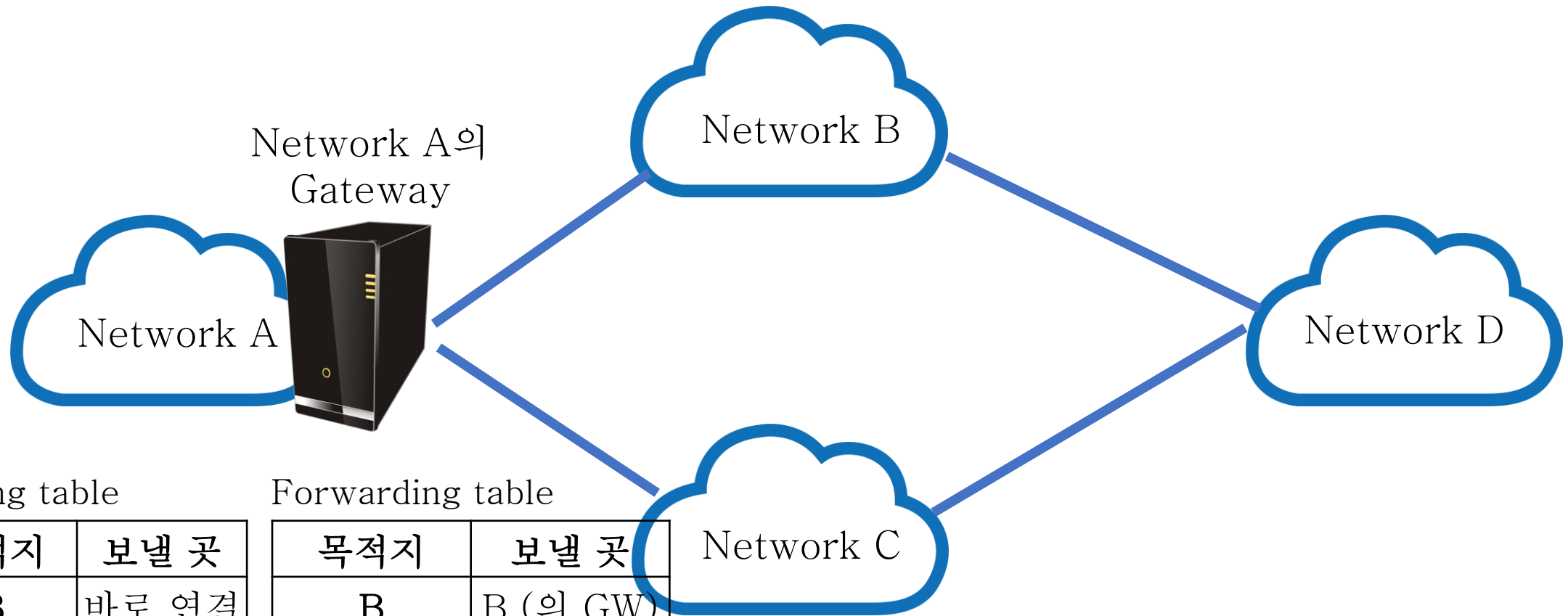
Gateway A 는 어떻게 B 로 보내는 선택을 할까?



Routing 과 Forwarding

- Routing
 - Route + -ing
 - 길찾기: 내가 어떤 선택가능한 경로들을 가지고 있나?
 - 데이터 전송의 본질적인 부분이 아닌, 컨트롤의 영역 (control plane)
- Forwarding
 - Forward + -ing
 - 전송하기: 선택 가능한 경로 중 하나를 골라서 보낸다.
 - 데이터 전송의 본질적 영역 (data plane)
 - 목적지에 따라서 골라서 보내기 때문에 Switching 이라고도 함

Routing 과 Forwarding



Routing table

목적지	보낼 곳
B	바로 연결
C	바로 연결
D	B 또는 C

Forwarding table

목적지	보낼 곳
B	B (의 GW)
C	C (의 GW)
D	B (의 GW)

Domain 간 사용할 경로 결정

- Domain: 서로 다른 관리 주체의 네트워크
또는 Autonomous System(AS)
- 관리 주체가 다른 도메인 간의 트래픽 이동 경로 선택
 - 단순히 빠르다고 선택하지 않음
 - 미국이 EU 와 통신하고 싶은데 북한을 경유하고 싶을까?
 - 빠르다고 하더라도 가격이 너무 비싸다면?
- Inter-domain 간 routing 은 “정책 기반”(policy-based)
 - 즉 단순히 속도, hop 수 등의 수치 (metric) 을 따르지 않고
정책이 그것을 override 하게 함

Routing 에서의 Lesson

- 인터넷은 복잡하게 얽혀 있으므로,
통신 중간중간에도 서로 다른 경로가 선택될 수 있다.
 - 우리가 개발하는 서비스의 latency/jitter 에 영향을 미친다.
- 특히 inter-domain 간의 통신은 빠르다고 선택되는 것이 아니라
주어진 정책에 가장 부합하는 경로를 선택한다.
 - Latency 가 지리적 거리와 어느 정도 부합하지만 꼭 그런 것은 아니다

Latency vs. Jitter

- Latency (Network delay, A.K.A 딜레이)
 - 송신자가 보낸 데이터가 얼마만에 수신자에게 도달하는가?
- Jitter
 - Latency 의 변화 정도

Latency vs. Jitter

명령 프롬프트

```
C:\Users\dkmoo>ping www.google.com
```

```
Ping www.google.com [172.217.161.228] 32바이트 데이터 사용:
```

```
172.217.161.228의 응답: 바이트=32 시간=36ms TTL=56
```

```
172.217.161.228의 응답: 바이트=32 시간=36ms TTL=56
```

```
172.217.161.228의 응답: 바이트=32 시간=37ms TTL=56
```

```
172.217.161.228의 응답: 바이트=32 시간=38ms TTL=56
```

```
172.217.161.228에 대한 Ping 통계:
```

```
패킷: 보냄 = 4, 받음 = 4, 손실 = 0 (0% 손실),
```

```
왕복 시간(밀리초):
```

```
최소 = 36ms, 최대 = 38ms, 평균 = 36ms
```

```
C:\Users\dkmoo>
```

Latency

Jitter 계산

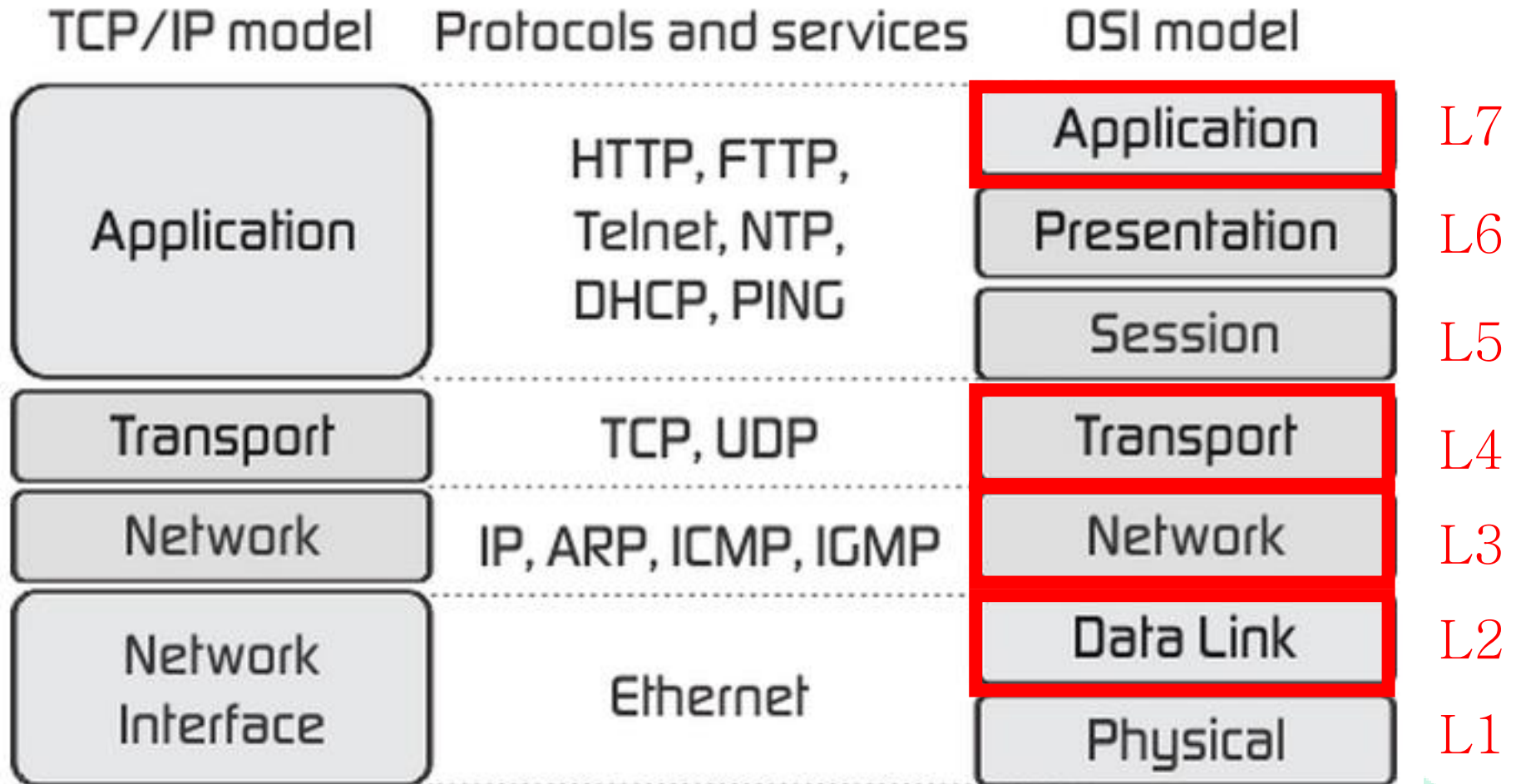
- Latency data points 간의 차이에 대한 평균값으로 계산
- Latency 가 36, 36, 37, 38 msec 인 경우
 - $|36 - 36| = 0$
 - $|36 - 37| = 1$
 - $|37 - 38| = 1$
 - $(0 + 1 + 1) / 3 = 0.67 \text{ msec}$

Latency vs. Jitter

- Interactive 한 서비스에 미치는 영향
 - Latency: 반응이 늦게 온다는 느낌을 줌
 - Jitter: 자꾸 튕다는 느낌을 줌

Ethernet, UDP, TCP, IP

TCP/IP 에서의 OSI Model 적용



Switch

- 사전적 의미: 전환하는 것
- 네트워크에서 스위치는 “이쪽으로 가라”, “저쪽으로 가라” 트래픽의 방향을 전환하는 것
- 어디로 보낼지 판단하기 위해서는 “판단 근거”가 있어야 함
- 네트워크 스위치에서의 판단 근거는 “해당 계층의 헤더”가 됨
- 그리고 그 계층의 숫자를 붙여서 “L얼마 스위치” 라고 부른다.

연습 문제

Q. Ethernet frame 의 목적지 정보에 따라 출력 포트를 선택해서 트래픽을 전달하는 장비가 있다. 이 장비는 넓게 봐서 뭐라고 부르면 좋을까?

A. L2 switch

연습 문제

Q. IP Packet 의 목적지 정보에 따라 출력 포트를 선택하는 장비가 있다. 이 장비는 같은 목적지라도 해당 패킷의 출처(=송신자) 가 사내인지 아니면 사외인지에 따라 다시 출력 포트를 조정한다. 이 장비는 넓게 봐서 어떻게 부르면 좋을까?

A. L3 switch

부연. 방화벽 장비 혹은 침입 탐지 시스템 (IDS; Intrusion Detection System) 등이 여기에 해당한다. 그들이 크게는 L3 스위치 기능을 가지고 있는 것이라는 것에 주목할 것

연습 문제

Q. HTTP 요청은 이를 처리하는 동등한 역할의 서버를 여러 대 운용한다 (pooling 이라고 하고, HTTP server pool 이 있다고 한다) 한 서버만 과부하가 걸리는 것을 막기 위해서 이 앞단에 pool 내 서버들에 round-robin 방식으로 분산해주는 장비가 있다. 이는 크게 봐서 뭐라고 부를 수 있을까?

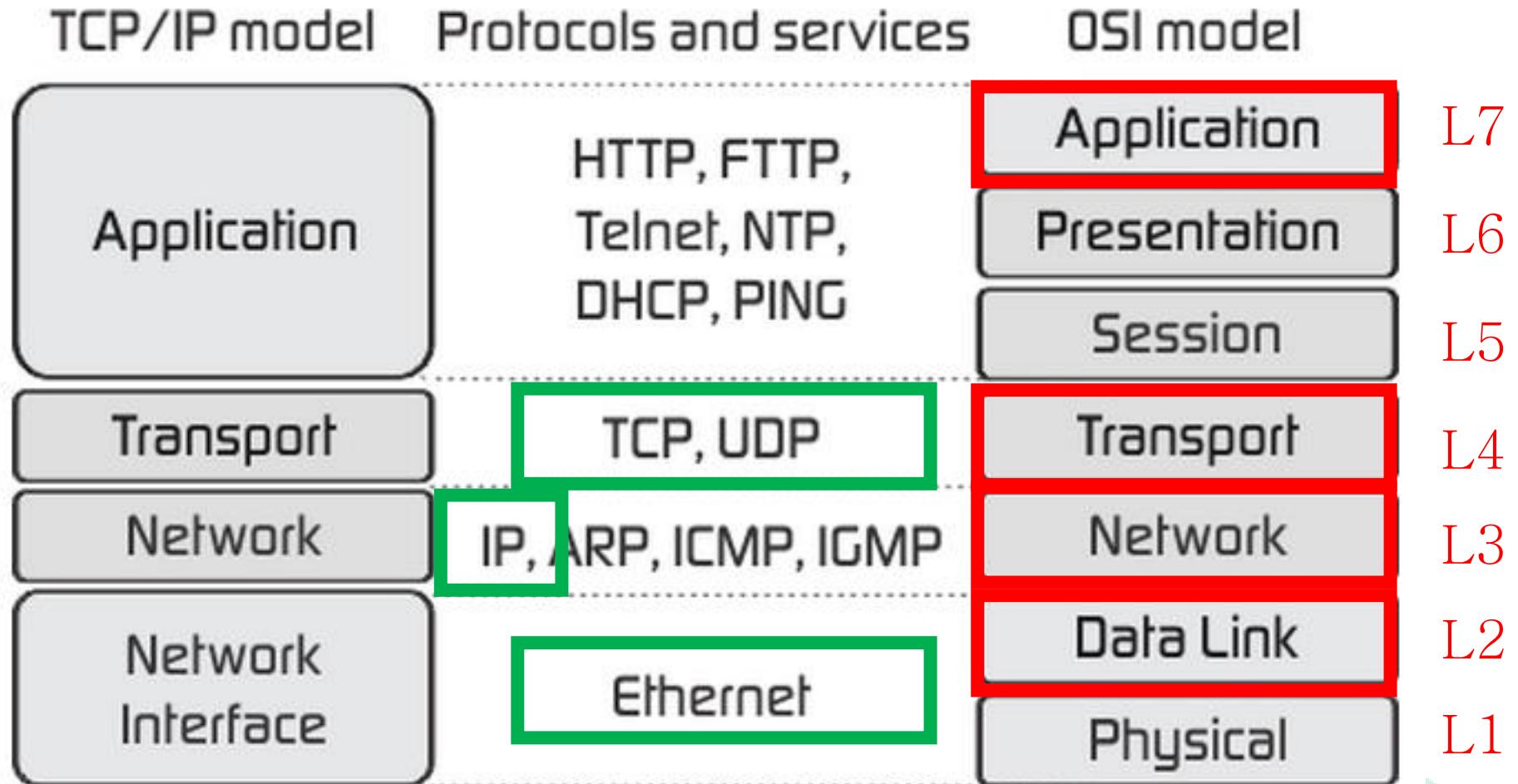
A. L7 switch

부연. Load balancer (LB) 가 여기 해당한다. LB역시 스위치이다. 그리고 밸런싱 기준 layer 를 명시해서 여기서는 L7 LB 이다.

Switch

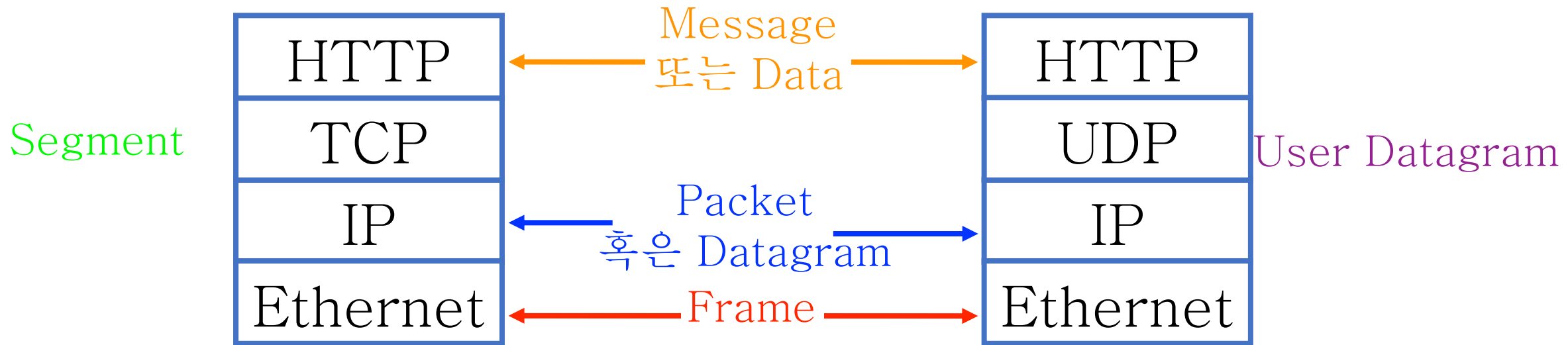
- Switch 는 기능을 의미하며 꼭 물리적인 장치를 뜻하지는 않는다.
- Cloud 에서는 대부분 소프트웨어로 이 기능을 제공한다.
 - L2/L3/L7에 따라 트래픽을 전달하는 소프트웨어
 - 예: AWS 에서 L7 에 따라 분배하는 기능을 Elastic Load Balancer (ELB) 라고 한다.

TCP/IP 에서의 OSI Model 적용



각 프로토콜의 1개 단위들

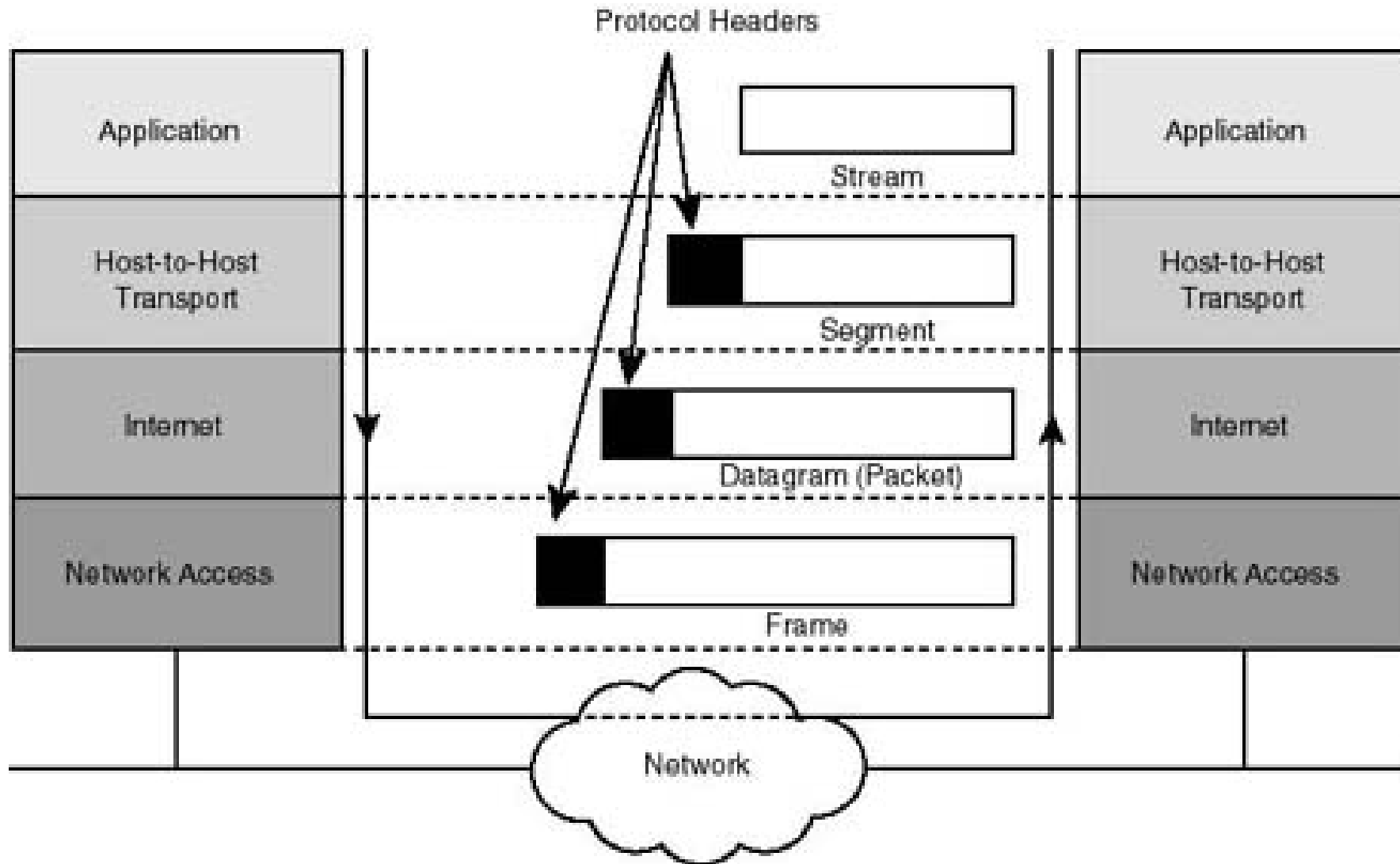
- Protocol Data Unit(PDU): 각 프로토콜의 1개 데이터를 지칭



Box-in-Box 기억해주세요

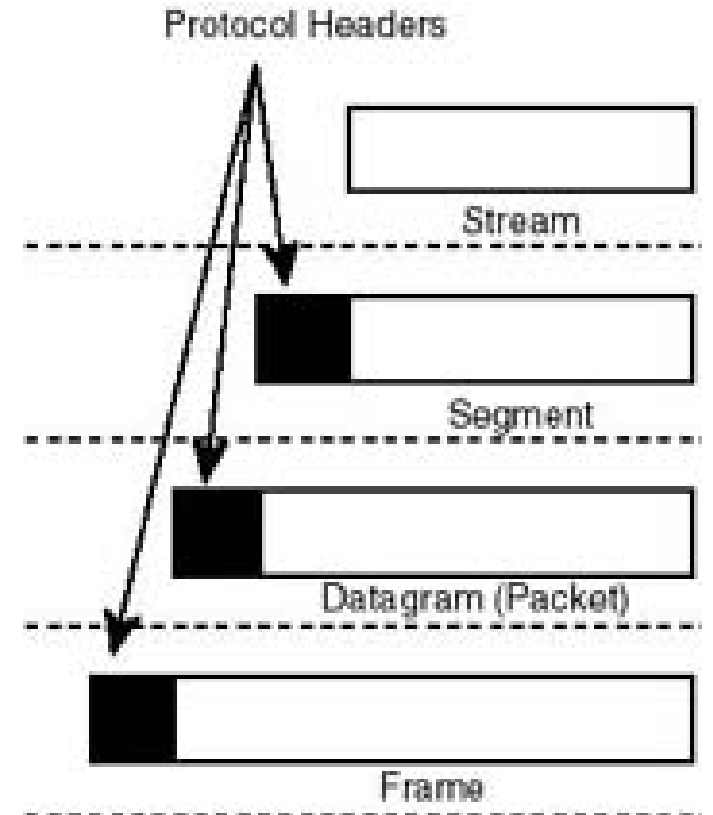


이 그림도 당연히 기억하시죠?



이 그림도 당연히 기억하시죠?

1. 검은색 부분은 헤더 (header).
즉, 각 프로토콜은 “헤더” + “데이터” 형태.
2. 헤더에 포장 박스에 쓴다고 한
송신자 주소, 수신자 주소가 들어감.
각 계층별 주소가 MAC, IP, Port 임.
3. 택배 박스 송장 스티커에 주소 말고도
바코드, 취급점 등등 부가 정보가 들어가듯이
헤더에는 주소 말고도 다른 정보도 들어감.



질문

Q. 프로토콜이 할 수 있는 것과 할 수 없는 것은 무엇에 의해 결정될까?

A. 헤더

Data Link Layer: Ethernet Frame

Q. 헤더 구조를 봤을 때

Ethernet 은 frame 오류를 걸러낼 수 있을까?

A. CRC 가 있으므로 가능함

Preamble	Start of frame delimiter	MAC destination	MAC source	802.1Q tag (optional)	Ethertype (Ethernet II) or length (IEEE 802.3)
7 octets	1 octet	6 octets	6 octets	(4 octets)	2 octets

Data (46–1500 bytes)...	Frame check sequence (32-bit CRC)	Interpacket gap
	4 octets	12 octets

Data Link Layer: Ethernet Frame

Q. 헤더 구조를 봤을 때

최대 보낼 수 있는 길이는? 그걸 넘으면 어떻게 될까?

A. 최대 길이가 정해져 있음. (일반적으로 최대 데이터는 1500B)

그걸 넘으면 버려지므로 상위 계층에서 쪼개서 보내야함.

Preamble	Start of frame delimiter	MAC destination	MAC source	802.1Q tag (optional)	Ethertype (Ethernet II) or length (IEEE 802.3)
7 octets	1 octet	6 octets	6 octets	(4 octets)	2 octets

Data (46–1500 bytes)...

Frame check sequence (32-bit CRC)	Interpacket gap
4 octets	12 octets

Network Layer: IP Packet(Datagram)

Q. IP packet 의 최대 길이는 얼마인가?

A. Total length 가 16bits 이므로 header 를 포함해서 64KB

IPv4 Header Format

0								1								2								3							
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
Version				IHL				DSCP						ECN		Total Length															
Identification																Flags				Fragment Offset											
Time To Live								Protocol								Header Checksum															
Source IP Address																															
Destination IP Address																															
Options (if IHL > 5)																															

Network Layer: IP Packet(Datagram)

Q. 64KB 보다 큰 데이터를 보내려면 어떻게 해야 될까?

A. 상위 계층에서 64KB 단위로 쪼개서 보내야 한다.

IPv4 Header Format

0								1								2								3							
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
Version				IHL				DSCP						ECN		Total Length															
Identification																Flags				Fragment Offset											
Time To Live								Protocol								Header Checksum															
Source IP Address																															
Destination IP Address																															
Options (if IHL > 5)																															

Network Layer: IP Packet(Datagram)

Q. Ethernet 에서 data 의 최대가 1500B 라고 했는데
어떻게 64KB까지 담을 수 있을까?

A. IP packet 하나가 여러 Ethernet frame 으로 쪼개진다. (=fragmentation)

IPv4 Header Format

0								1								2								3							
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
Version				IHL				DSCP						ECN		Total Length															
Identification																Flags		Fragment Offset													
Time To Live								Protocol								Header Checksum															
Source IP Address																															
Destination IP Address																															
Options (if IHL > 5)																															

Network Layer: IP Packet(Datagram)

Q. IP 는 에러를 검출할 수 있을까?

A. Header 영역의 에러만 제한적으로 검출할 수 있다.

IPv4 Header Format

0								1								2								3							
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
Version				IHL				DSCP						ECN		Total Length															
Identification																Flags		Fragment Offset													
Time To Live								Protocol								Header Checksum															
Source IP Address																															
Destination IP Address																															
Options (if IHL > 5)																															

Network Layer: IP Packet(Datagram)

Q. IP 는 왜 header 의 오류만 체크할까?

A. 효율성 때문임. Hop-by-Hop 방식인 IP 는 hop 을 지날 때마다 (= gateway 를 거칠 때마다) header 필드 중 TTL 을 변경함. 즉, checksum 은 무조건 변경됨. 따라서 전체 를 매번 다시 계산하는 것이 부담됨.

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
Version				IHL				DSCP						ECN		Total Length															
Identification															Flags			Fragment Offset													
Time To Live								Protocol						Header Checksum																	
Source IP Address																															
Destination IP Address																															
Options (if IHL > 5)																															

Network Layer: IP Packet(Datagram)

Q. IP 는 전송을 보장할까?

A. Header 에 그런 field 가 없다. 그러니 안 한다.

IPv4 Header Format

0								1								2								3							
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
Version				IHL				DSCP						ECN		Total Length															
Identification																Flags				Fragment Offset											
Time To Live								Protocol								Header Checksum															
Source IP Address																															
Destination IP Address																															
Options (if IHL > 5)																															

Network Layer: IP Packet(Datagram)

- 64KB 까지 (헤더 포함) 한 번에 보내고 한 번에 받을 수 있다.
 - 100 bytes 를 보내면 받는 쪽에서 정확히 100 bytes 를 읽을 수 있다.
- 그러나 하위 data link layer 의 payload 길이 제한을 넘는 경우 여러 개의 IP packet 으로 쪼개져서 전송된다.
- 받는 쪽에서는 쪼개진 data link layer 의 payload 를 재결합해서 IP packet 으로 복원한다.
- 이 때 하나라도 누락이 되면 IP packet 은 전체 누락된다.
 - 누락 이유: Congestion, bit error
- IP 는 최선을 다해 전송을 시도할 뿐 전송을 보장하지는 않는다.
이를 best-effort 라고 한다.

Transport Protocols (L4 Protocols)

- User Datagram Protocol (UDP)
- Transmission Control Protocol (TCP)

Transport Layer: UDP Datagram

Q. UDP 헤더를 볼 때 IP 위에 어떤 부가 기능을 제공하는가?

A. 포트를 구분하는 것, checksum 외에 IP 속성 그대를 갖게 된다.

UDP Header																															
0								1								2								3							
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
Source port																Destination port															
Length																Checksum															

Transport Layer: UDP Datagram

- IP 와 마찬가지로 datagram 이라는 이름을 사용한다. IP 와 구분하기 위해서 User 를 붙여서 User Datagram 이라고 부른다.
- Port 를 제공하고 checksum 계산하는 것 외에 특별한 기능이 없다. 이는 L3 layer (network layer) 인 IP 의 기능을 L4 (transport layer) 에서 그대로 가져다 쓰게 하기 위해서 만들어진 프로토콜이기 때문이다.
- IP 의 속성을 갖기 때문에 IP 헤더 포함 64KB 이내에서는 보낸 그대로 수신 측에서 받게 된다.
 - 100 bytes 를 보내면 받는 쪽에서 read 할 때 100 bytes 를 읽음
- 독자적으로 쪼개고 재결합하는 기능을 제공하지 않는다. (IP 에 의존함)
- IP 와 마찬가지로 best-effort 이다.

Transport Layer: TCP Segment

- 특징1: 데이터가 연속적인 것 같은 환상을 지원한다. (stream)
- 특징2: 받았는지 안 받았는지 체크한다.
- 특징3: 수신자의 여력을 체크한다.

TCP Header																															
0								1								2								3							
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
Source port																Destination port															
																Sequence number															
																Acknowledgment number (if ACK set)															
Data offset		Reserved 000		NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size																		
Checksum																Urgent pointer (if URG set)															
Options (if <i>data offset</i> > 5. Padded at the end with "0" bytes if necessary.)																															
...																															

TCP: 연속되는 Data Stream

- TCP 의 sequence 번호 (그리고 ACK 번호)는 전송되는 데이터를 byte 단위로 트래킹한다.
- IP packet이 헤더 포함 64KB 까지이므로 TCP 는 전송하는 데이터 양에 따라 IP packet 여러 개로 쪼개서 전송한다.
- 수신 측에서는 여러 IP 에 걸쳐 있는 TCP segment 를 sequence 번호를 이용해 결합해서 복원한다.
- 이 때문에 TCP 는 하나씩 끊겨서 전송되는 IP 위에서 마치 data 가 연속으로 흐르는 것 같은(stream) 환경을 지원할 수 있다.

TCP: Connection 기반

- 앞의 설명에서 보듯 TCP 는 특정 상대방을 가정하고 전송한 data 의 byte 단위 sequence 번호, 어디까지 수신했는지 byte 단위의 ack 번호를 관리한다.
- 이렇게 통신 양 끝단(end-point) 의 state 를 관리하기 때문에, TCP 는 stateful 하고 connection 기반이 된다.

TCP 의 전송 제어

- TCP 의 전송 제어가 신경 쓰는 것
 - 수신자의 여력 (버퍼크기)
 - ➔ TCP header 중 Window Size 필드
 - 중간에 거쳐가는 게이트웨이(라우터) 들의 여력 (버퍼크기)
 - ➔ 받았는지 여부 (= TCP header 중 SEQ/ACK 필드)

TCP Window Size

- 상대방에게 자신이 얼마나 큰 데이터를 한 번에 받을 수 있는지 advertise 함 (receive window size 라고 함)
- Socket library 로는 setsockopt 로 변경 가능
- 그러나 여기서 advertise 된 window size 단위로 데이터가 전송되지 않고, 뒤에 설명하는 혼잡 제어 알고리즘에 따른 window 크기와 비교해서 둘 중 작은 사이즈로 데이터를 전송한다.

Network Congestion (네트워크 혼잡)

- 네트워크 경로상의 장비나 회선이 과부하 걸려서 traffic 을 drop 하는 것

TCP의 혼잡 제어 (Congestion Control)

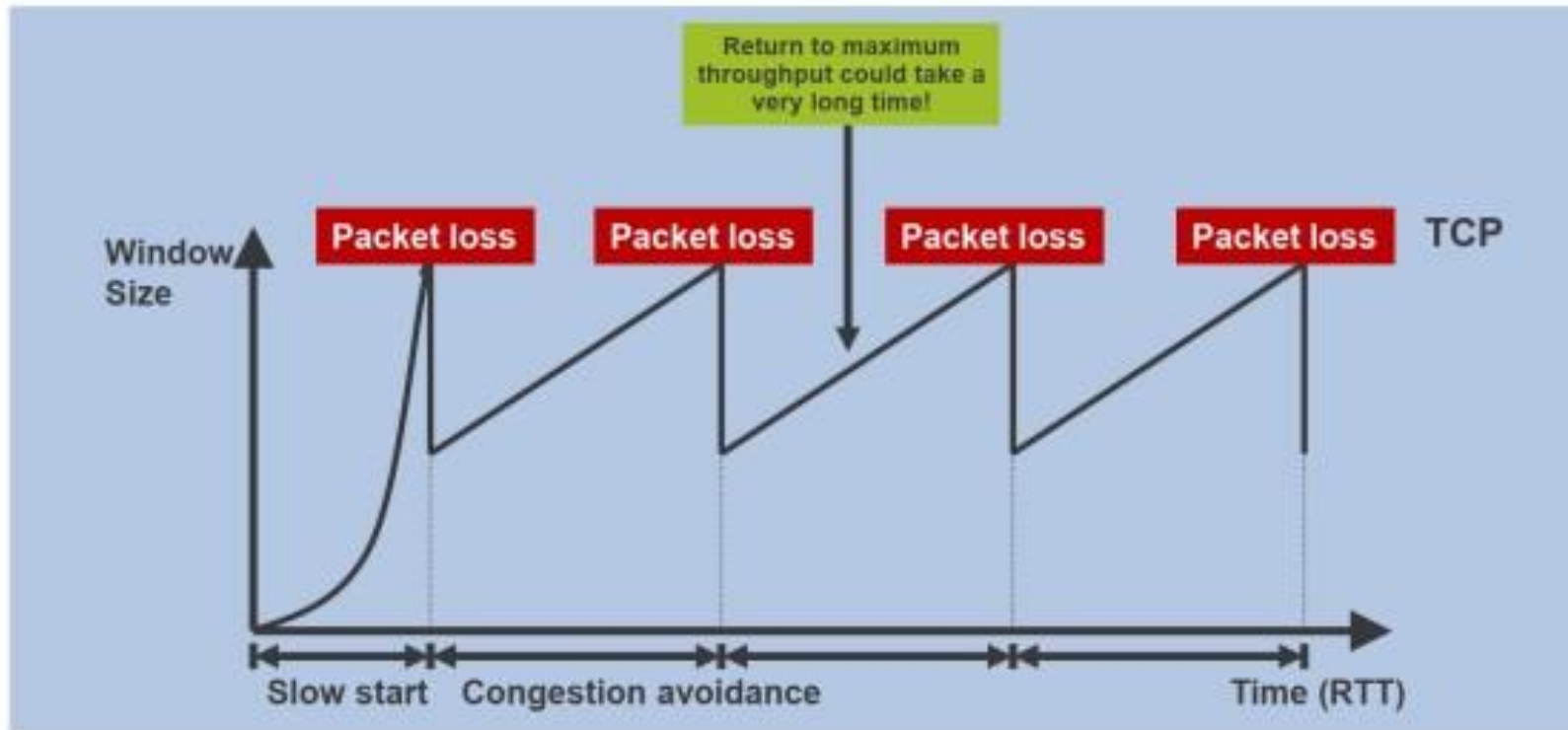
- 수신자는 data 를 받으면
송신자에게 “다음에 보낼 SEQ 를 의미하는 ACK” 를 전송함
- 송신자는 1byte 보내고 ACK 를 기다리고 이렇게 동작하면 너무 비효율적이
니 한번에 보낼 수 있는 데이터 양만큼 보내고 ACK 를 기다림
- 이 때 ACK 를 받을 때마다 이 “한번에 보낼 수 있는 데이터 양” 를 증가시킴
(ACK 가 잘 오면, 1B, 2B, 4B, 8B, 이런 식으로...)
- 그러다가 ACK가 누락되거나 순서가 뒤집혀 오면 혼잡이라고 판단하고 1)한
번에 보낼 수 있는 데이터 크기를 리셋하고, 2) 이전 보낸 데이터를 재전송함
- 이 “한번에 보낼 수 있는 데이터 크기”라는 것을 congestion window 라고 함.

TCP Congestion Control 의 영향

- 처음부터 full speed 로 데이터를 전송하지 않는다.
(congestion window 를 ACK 에 따라 증가시키므로 전송 속도가 점점 증가한다.)
- Congestion 이라고 생각하면 “이크!” 하면서 움츠러들고 congestion window를 줄인다. 따라서 전송 속도가 줄어든다.
- 그런데 congestion 이 발생하지 않으면 congestion 이 발생할 때까지 계속 congestion window를 올려가기 때문에 TCP 의 전송 속도는 일정하지 않고 오락가락 하게 된다.

TCP Congestion Control 의 영향

TCP Behavior



RFC1323 - TCP Extensions for High Performance

TCP Congestion Control 의 영향

Q. TCP 로 100B 를 보내려고 한다. 몇 개의 IP 패킷으로 갈까?

A. 모른다.

Congestion window 크기가 1B인 상태에서는

$1B + 2B + 4B + 8B + \dots$ 이렇게 여러 개로 쪼개서 갈 것이다.

그러나 window가 100B 이상인 경우는 한번에 갈 것이다.

TCP Congestion Control 의 영향

Q. 송신 측에서 100B 를 보냈다.

수신 측에서 100B 를 읽으려고 하면 100B 가 읽힐까?

A. 앞의 질문의 답에 따라 그럴 수도 있고 적게 읽힐 수도 있다.

이 때문에 TCP socket 프로그래밍에서는 이를 반드시 고려한다.

- 즉, 받기 관련된 함수들이 요청한 바이트 수만큼 받고 반환하는 것이 아니다.

UDP vs. TCP

- UDP 와 TCP 는 모두 L4 프로토콜이다.
- UDP 는 L3 인 IP 를 L4 에서 쓸 수 있게 하기 위한 껍데기다.
- TCP 는 혼잡 제어와 그 과정 중에 재전송이라는 기능을 제공한다.

UDP vs. TCP

Q. 어떤 쪽이 더 빠른가? 그 이유는?

A. UDP 는 요청한 datagram 을 즉각적으로 최선을 다해 전송한다.
반면 TCP 는 서서히 속도를 올린다.

따라서 같은 사이즈의 데이터를 전송할 때 UDP 가 빠를 수 있다.

더구나 TCP 는 누락되는 것을 기다려 재전송한다.
이 때문에 속도는 더 느릴 수 있다.

UDP vs. TCP

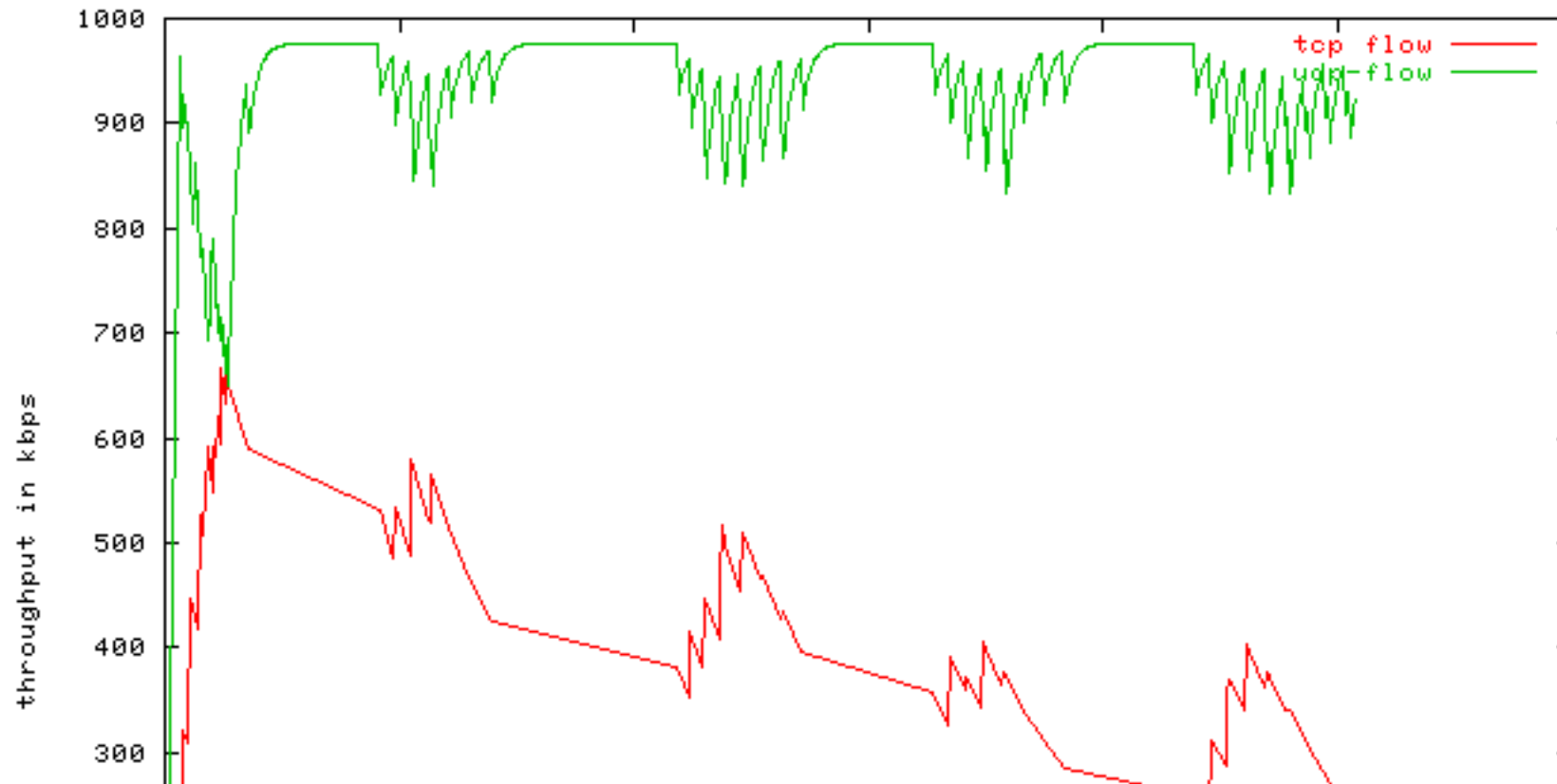
- 따라서 필요한 응용에 따라 둘의 선택은 달라진다.
- 누락 되어도 상관없고 가급적 빨리 전송해야 되는 것 → UDP
 - 일반적으로 interactive 한 audio, video 등
 - 누락되는 것을 대비해서 각각의 datagram 에 충분한 데이터를 넣는다.
그래서 앞의 것이 누락 되어도 뒤의 것을 받고 복구할 수 있게끔
- 누락되면 안되는 데이터를 안정적으로 전송해야 되는 것 → TCP
 - file 전송, 인증 처리 등

연습 문제

Q. 호스트가 1Gbps L2 스위치에 연결되어 있다.
호스트에서 UDP 소켓 1개와 TCP 소켓 1개를 열고
각각 보낼 수 있는 최고 속도로 데이터를 전송하기 시작한다.
이 때 steady-state 에 도달하는가?
만약 그렇다면 각각의 소켓이 차지하는 대역폭은 대략 얼마인가?

UDP vs. TCP

- TCP 는 패킷 유실을 경험하면 네트워크 과부하로 판단한다.
- 그래서 congestion window 를 줄이고 이는 전송 트래픽을 줄이게 된다.
- 그러나 UDP 는 과부하를 신경쓰지 않고 TCP 가 양보한 대역폭을 차지한다.



UDP vs. TCP Summary

- TCP
 - 전송 속도를 서서히 올린다.
 - 재전송을 제공한다.
 - 혼잡 제어를 통해 트래픽을 양보한다.
 - 위의 특성들 때문에 전송 속도는 진동한다.
 - 위의 특성들 때문에 end-to-end 의 state 를 관리하므로 연결 기반이다.
 - 속도 상관없이 안정적인 통신에 적합
- UDP
 - IP 와 마찬가지로 best-effort 다.
 - 안정성이 희생되더라도 빠른 통신에 적합
 - End-to-end 로 관리되는 state 가 없으므로 연결 기반이 아니다.

Summary

- 컴공에서 사용되는 많은 용어는 기존에 존재하는 것들에서 차용했다. 따라서 용어 자체는 목적이 아니라 수단이 되어야 한다.
- “인터넷”은 네트워크를 연결하는 한 방식이다.
- 인터넷에서 사용되는 TCP/IP 모델은 OSI7 모델의 단순한 형태이다.
- 이처럼 layering 은 단순화와 각 layer 의 독립적 발전을 제공한다.
- 인터넷의 layer 의 특성은 protocol header 로 결정된다.
- 절대 반지 같은 silver bullet protocol 은 없다.
그 속성을 잘 이해하고 있다가 상황과 목적에 따라 선택해야 된다.