DNS

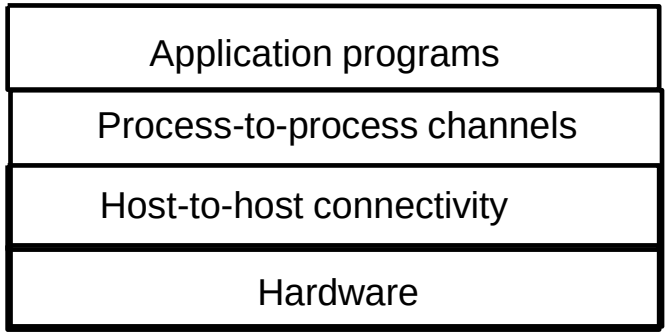
고재원

이해가 어렵거나 내용이 잘못 된 것 같으면
부담 없이 부부젤라를 불 것

목차

- 프로토콜, TCP/IP스택 복습
- 실제 통신 예시
- DNS

프로토콜?

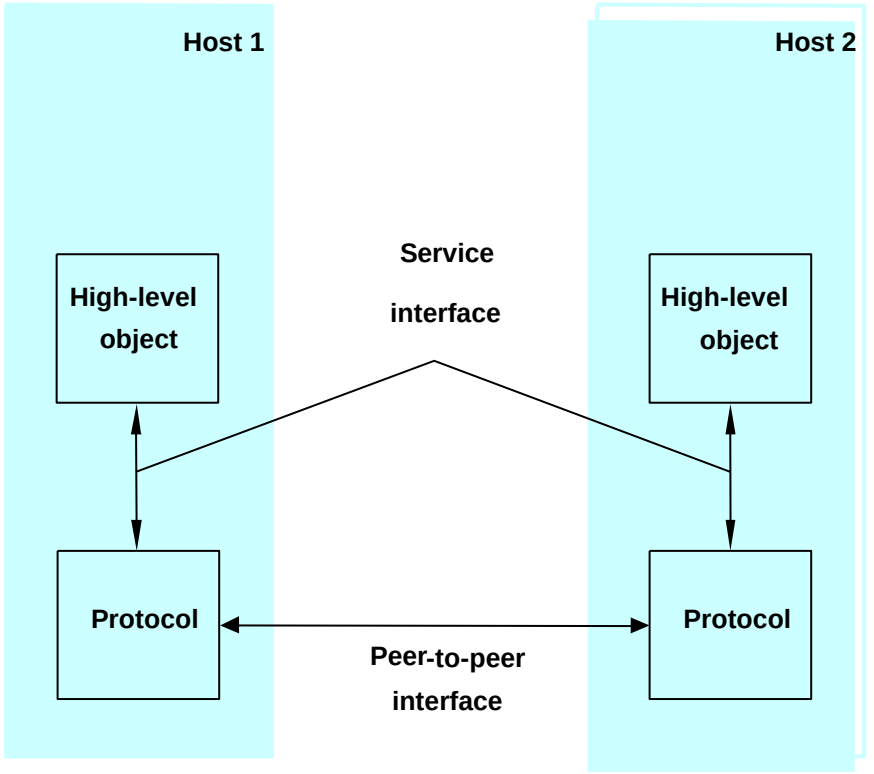


- 통신 주체 사이의 약속
- 수 많은 프로토콜, 매번 바뀌는 스펙 등으로 인한 혼란
- 구조적으로 해결해보자!
  - 추상화(짬 때리기)를 통한 구조화
  - 약속에 대한 계층이 만들어진다. (계속 밑으로 짬 때리니깐)
    - 각 계층은 하는 일이 분리가 된다.

프로토콜?

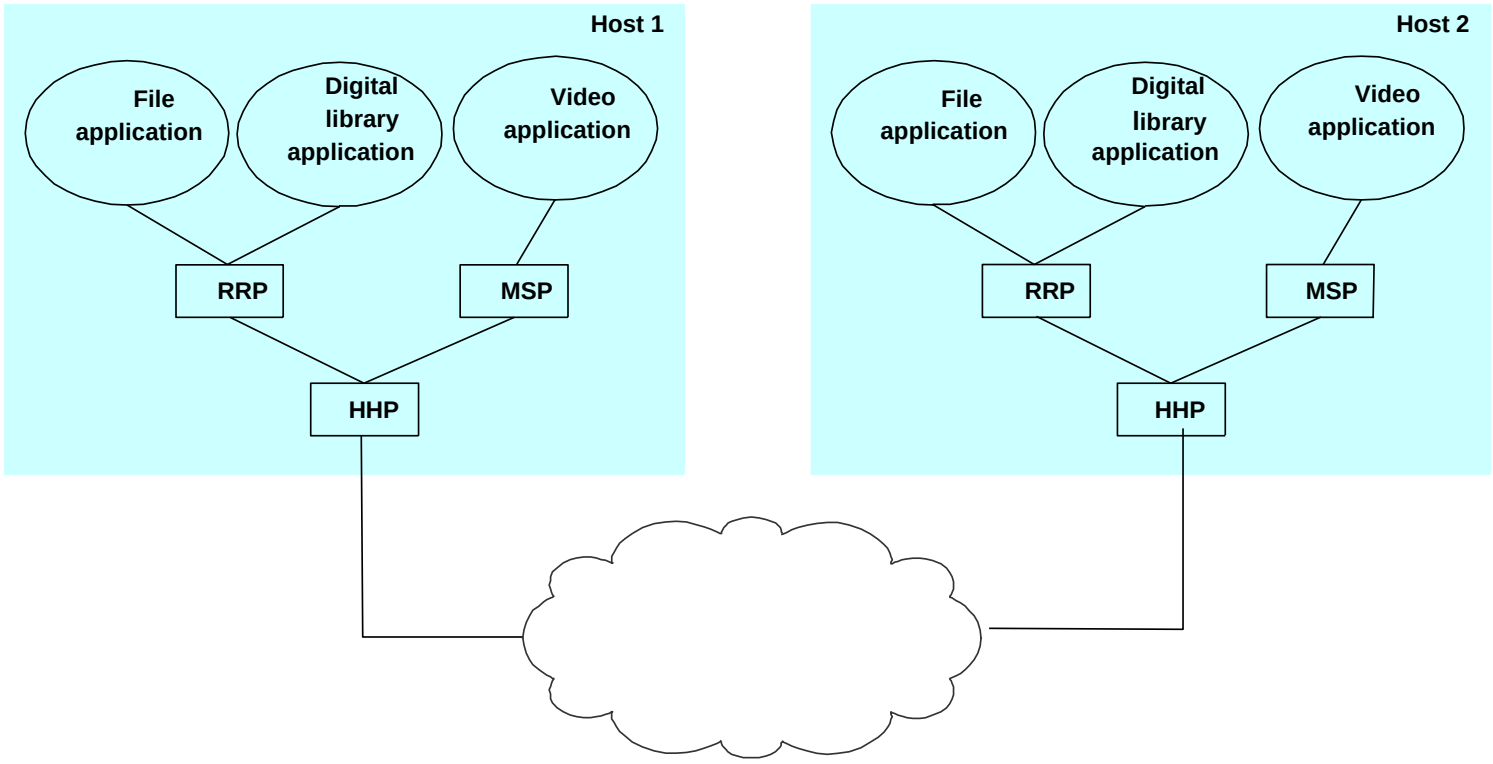
- 통신 주체 사이의 약속

- 동료(peer to peer) 인터페이스
  - 통신 대상의 같은 계층의 프로토콜과 주고 받을 메시지를 정의



- 서비스 인터페이스
  - 해당 프로토콜을 정의
  - 하위 계층의 프로토콜에게 내가 무슨 일을 하는지 알림

프로토콜?

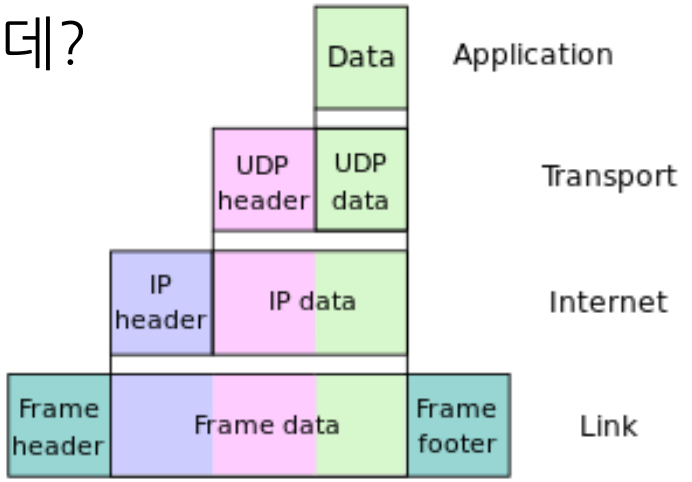


동료 간의 통신은 대개 **간접적**으로 이루어진다.

- 실제 통신은 **하위 계층을 사용**하여 (즉, 위임하여) 이루어진다
- 하드웨어 수준에서는 직접적으로 통신이 이루어진다.

프로토콜의 동작 원칙

그래. 프로토콜이 계층적으로 이루어진 것도 알았고,
같은 계층과 주고받는 메시지를 정의하고 하위 계층에 어떤 일을 하는지 정의하는 것도 알았어.
그래서 이 둘을 만족하면서 통신을 하려면 어떻게 해야 하는데?

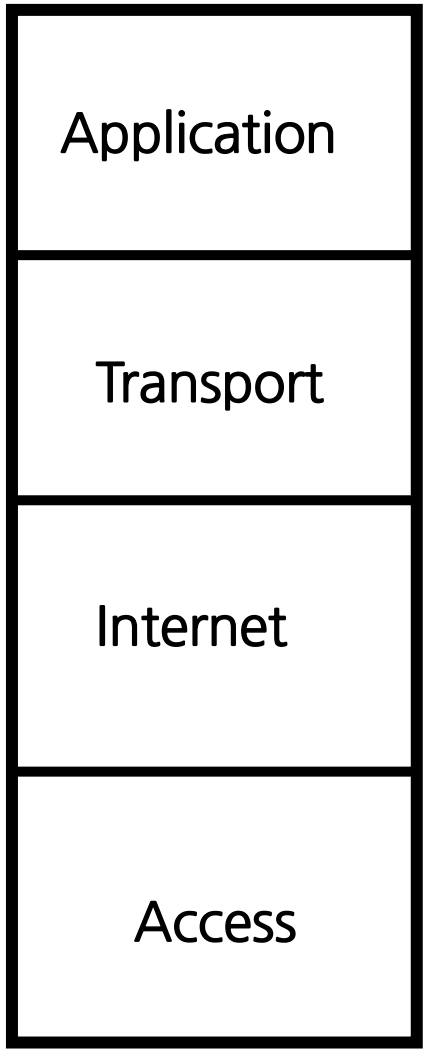


➔ 캡슐화

프로토콜은 데이터를 (헤더 + 바디)의 형태로 주고받는다.
즉

헤더 : 동료 인터페이스 (서로 어떤 데이터를 주고받을지)
바디 : 서비스 인터페이스 (내가 할 일이 무엇인지)

TCP/IP 스택 엑 인 트 어



프로세스에서 **유저 데이터**를 만들고 다른 프로세스와 이 데이터를 주고 받는다.

프로세스간의 채널(통로)를 열어 메시지 교환 담당

연결된 단말기 사이의 데이터를 주고받는다.

물리적인 신호를 이용하여 데이터를 주고받는다.

TCP/IP계층에서 사용하는 프로토콜

Internet protocol suite

Application layer

BGP • DHCP (v6) • DNS • FTP • HTTP (HTTP/3) • HTTPS • IMAP • IRC • LDAP • MGCP • MQTT • NNTP • NTP • OSPF • POP • PTP • ONC/RPC • RTP • RTSP • RIP • SIP • SMTP • SNMP • SSH • Telnet • TLS/SSL • XMPP • *more...*

Transport layer

TCP • UDP • DCCP • SCTP • RSVP • QUIC • *more...*

Internet layer

IP (v4 • v6) • ICMP (v6) • NDP • ECN • IGMP • IPsec • *more...*

Link layer

ARP • Tunnels • PPP • MAC • *more...*

V • T • E

무진장 많다.

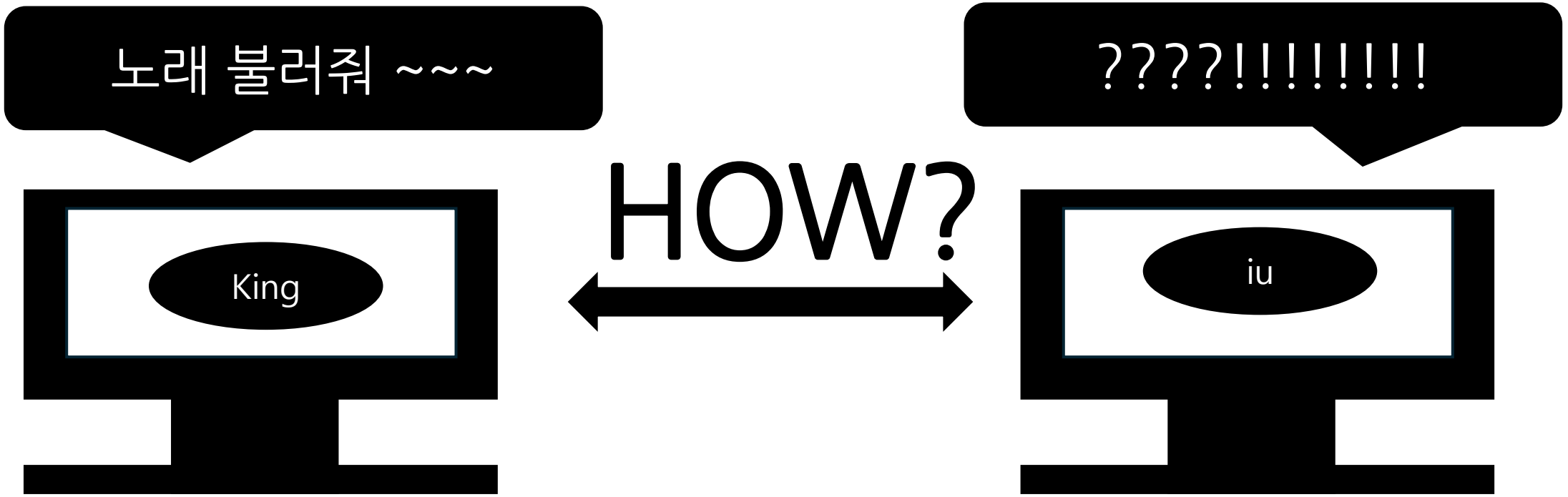
그러나?
각 계층에서 정의한 약속은 같다.

통신, 해보자2

지금까지 스터디를 통해 배운 것
그리고 스터디에서 다루지 않았던 것들을 이용하여 통신을 해 보자.

통신 목표

KBS의 컴퓨터의 king이란 웹 브라우저에서
HS_Nam의 컴퓨터의 iu 서버에 노래를 불러달라는 요청을 보내 보자.



1. King 브라우저에서 url을 입력한다.

King은 도메인 이름(url)은 알지만, 실제 이 프로세스가 어디에 있는지 정확히 모른다.

그런데 실제 통신을 하려면, 하위계층에서 실제 요청을 받을 프로세스가 어디에 있는지를 알아야 한다.

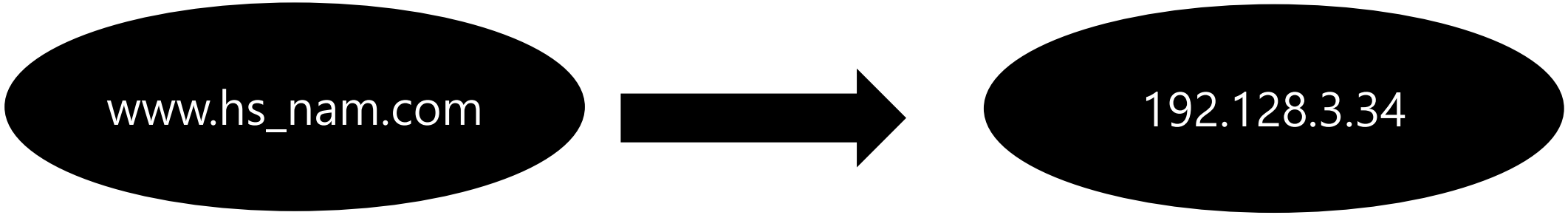
DNS(Domain Name System)

1. 인터넷에서는 인터넷에 연결된 호스트(컴퓨터)를 식별하기 위해 ip주소를 쓰는데, 이게 사람들에게 참 어렵다.

2. 따라서 사람들에게 익숙하지 않은 ip체계 대신 사람이 이해 할 수 있는 언어 형식으로 변환 한 것이다.

3. 통신시 컴퓨터를 위해 다시 ip로 변환해야 하는데, 이를 위해 DNS server가 있어 이 서버에 요청을 보내어 url로부터 ip를 얻을 수 있다. 이를 캐싱, 파일 저장하여 나중에 빠르게 접근한다.

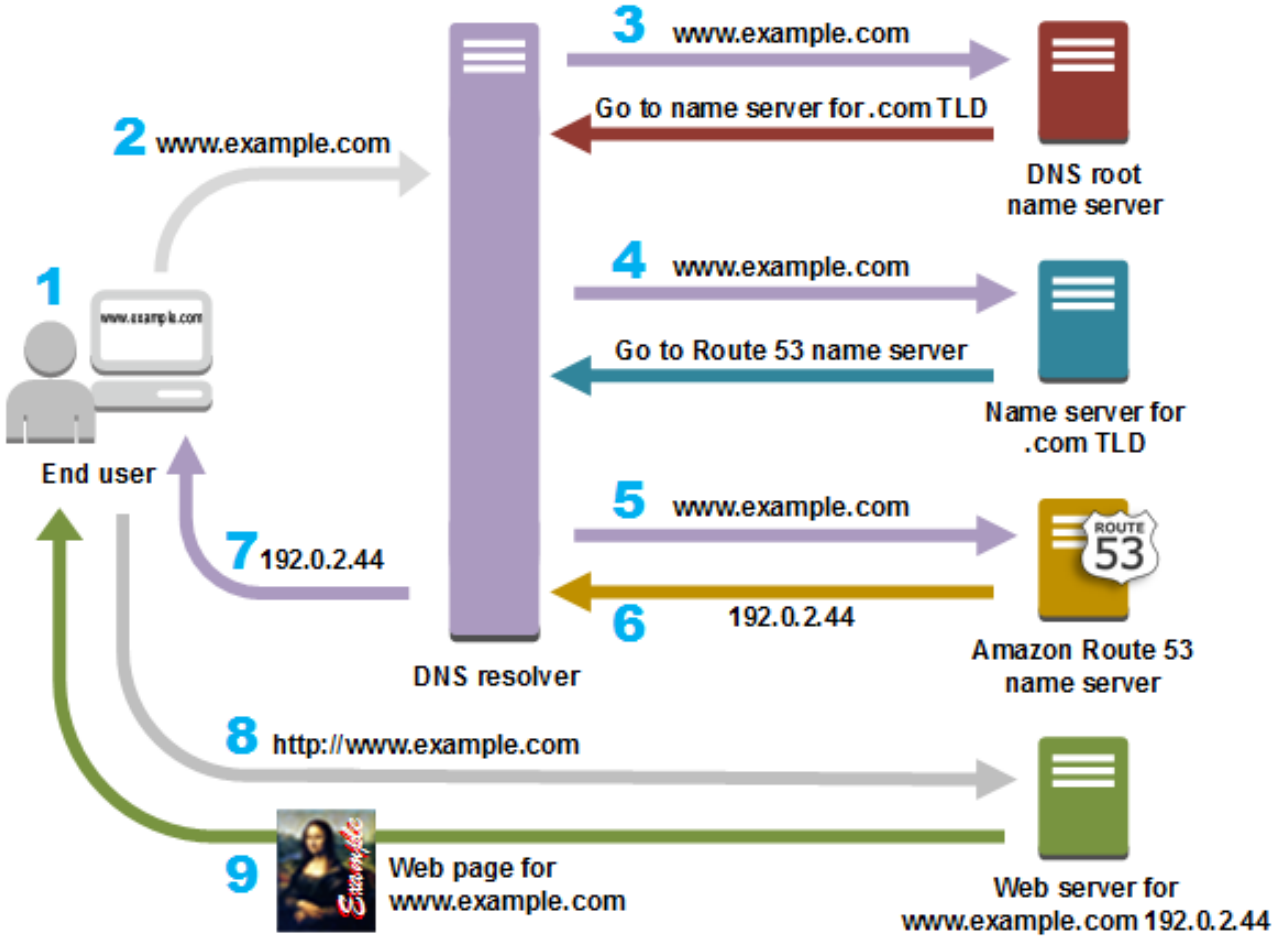
url에서 ip주소를 얻는 방법



1. 브라우저의 DNS cache를 살펴본다.
2. Hosts란 파일에서 찾아본다. c:\Windows\System32\drivers\etc
3. 1,2에도 없으면 DNS 캐시 서버로 DNS프로토콜을 이용하여 요청을 보낸다.

즉 url -> ip의 맵핑이다.

DNS서버의 작동 원리



DNS 서버는 분산 계층 구조를 가진다.

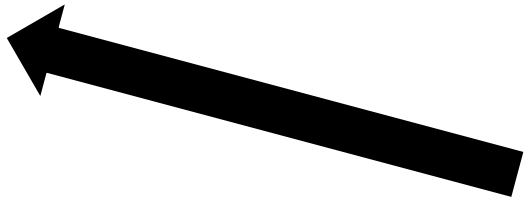
다음 예시를 보고 오면 그림이 이해가 된다.

DNS서버의 입장에서

www.naver.com

누군가가 요청을 보냈다.

www.naver.com 의
ip주소가 뭐에용?

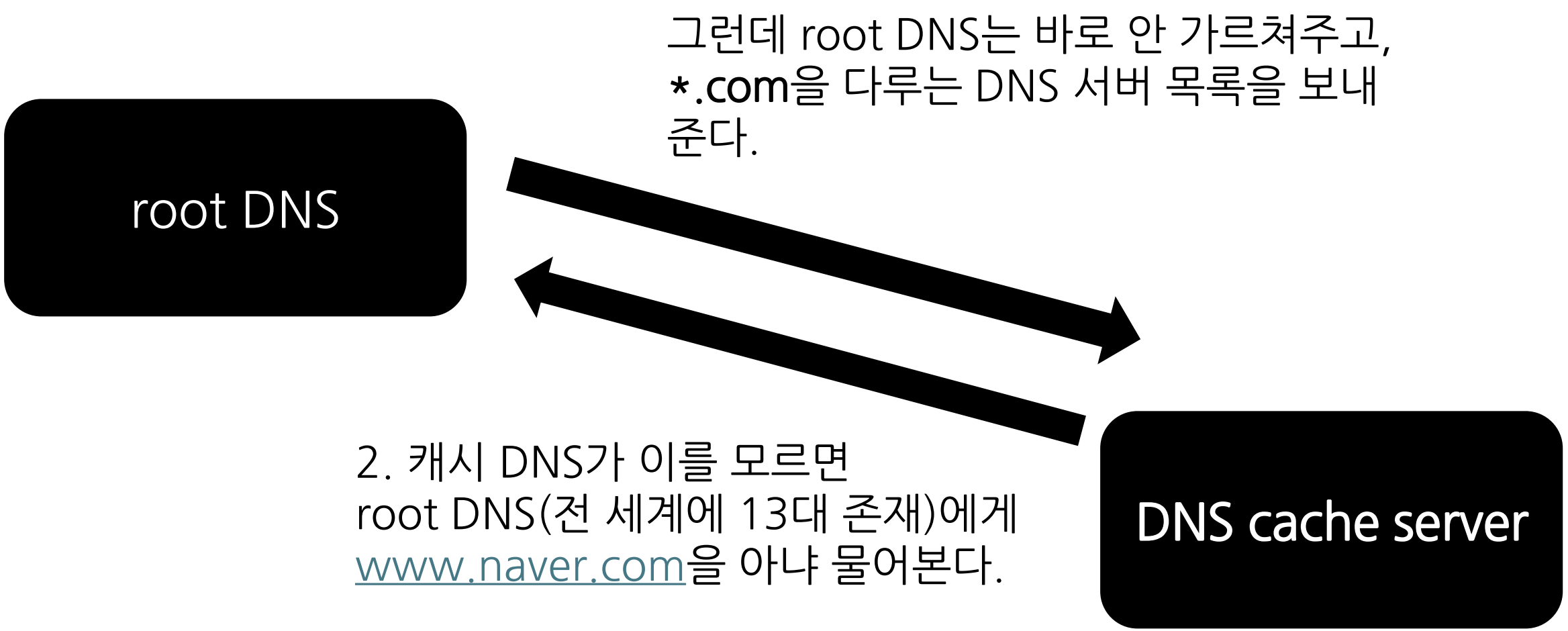


DNS cache server

1. 캐시 DNS가 이를 알고 있으면
바로 응답을 한다.

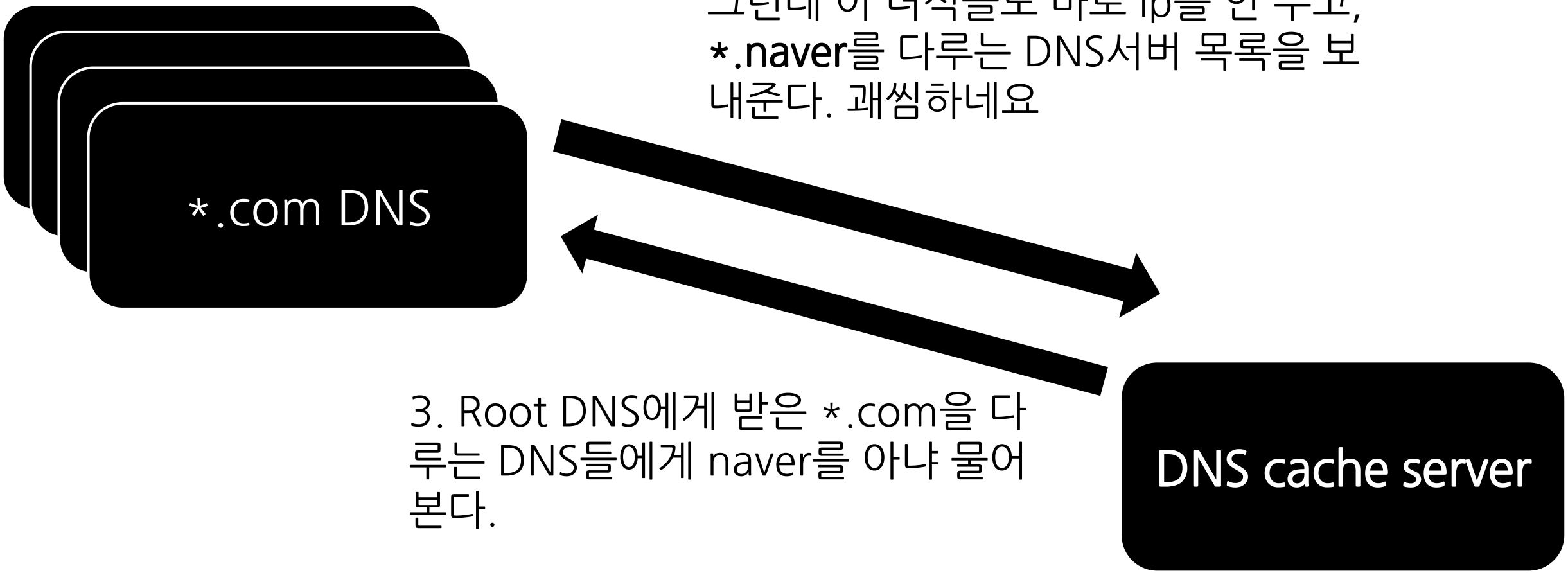
DNS서버의 입장에서

www.naver.com



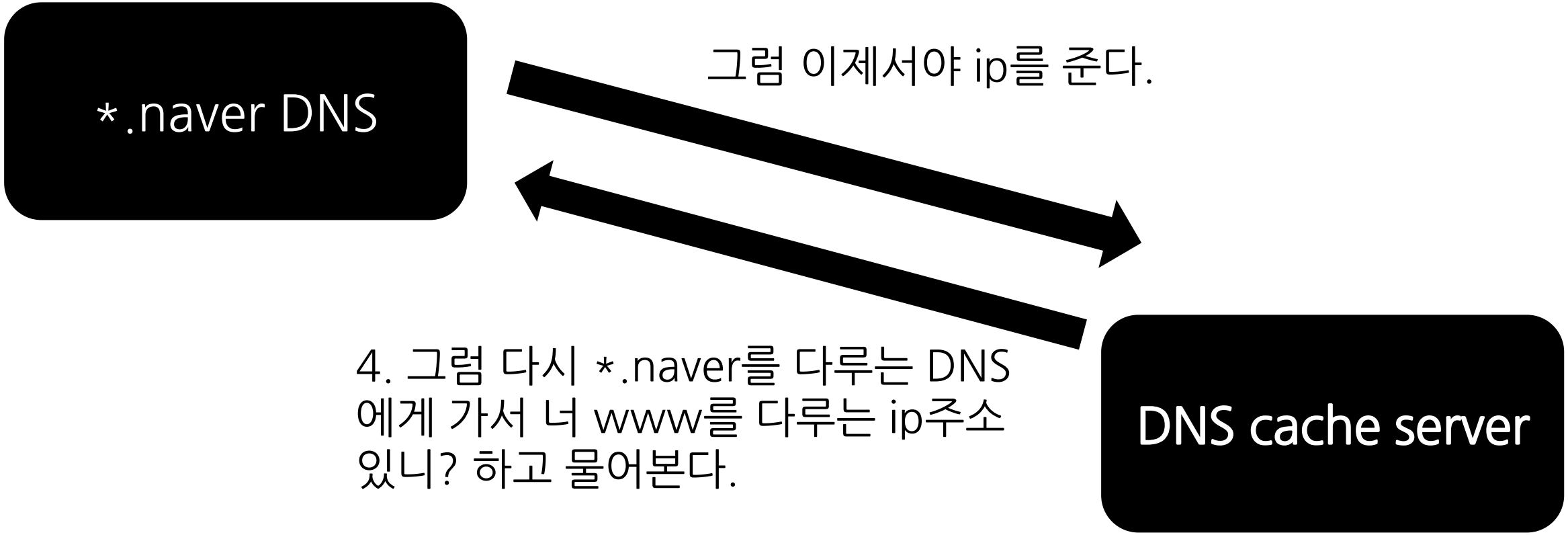
DNS서버의 입장에서

www.naver.com



DNS서버의 입장에서

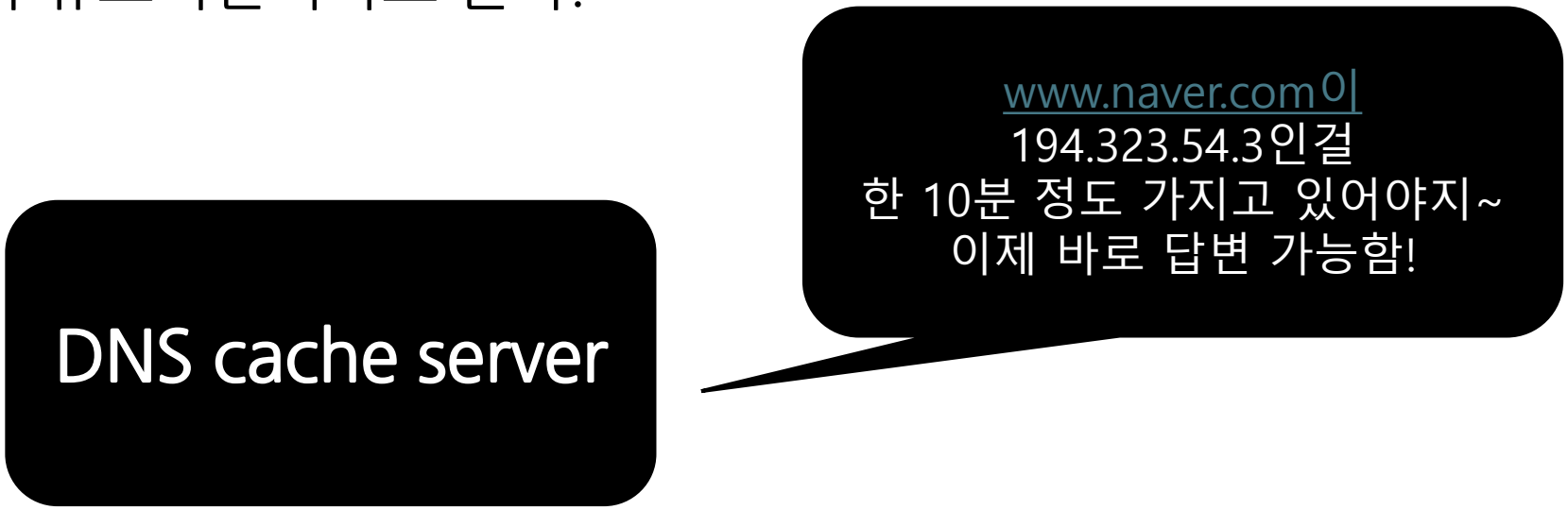
www.naver.com



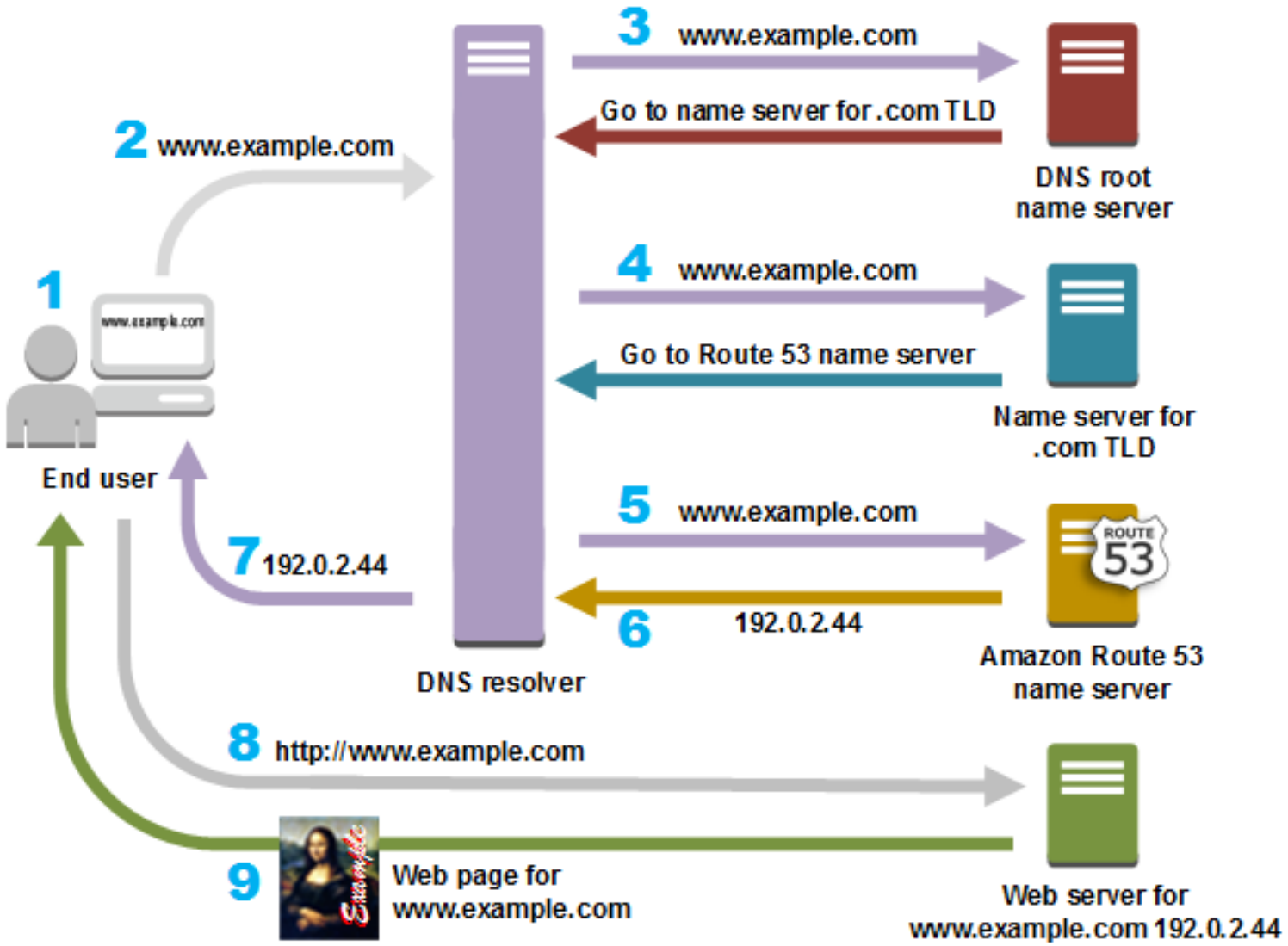
DNS서버의 입장에서

그런데 이걸 매번 물어볼 수 없으므로,
캐싱 서버에 일정 시간 동안 누가 똑같이 물어보면 저렇게 답해라 라고 한다.

그게 캐싱의 유효기간이라고 한다.

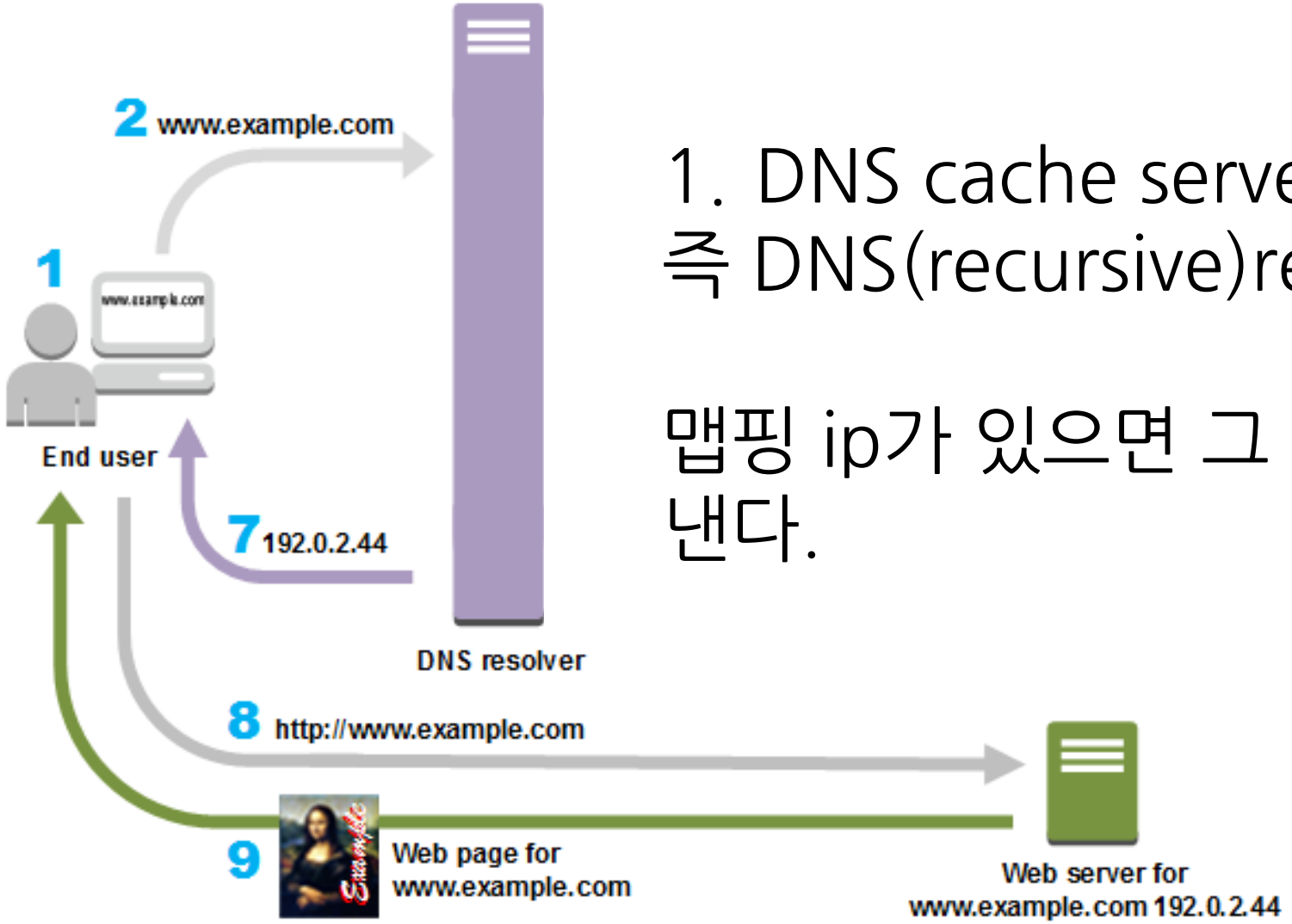


DNS서버의 작동 원리



다시 그림을 보자.

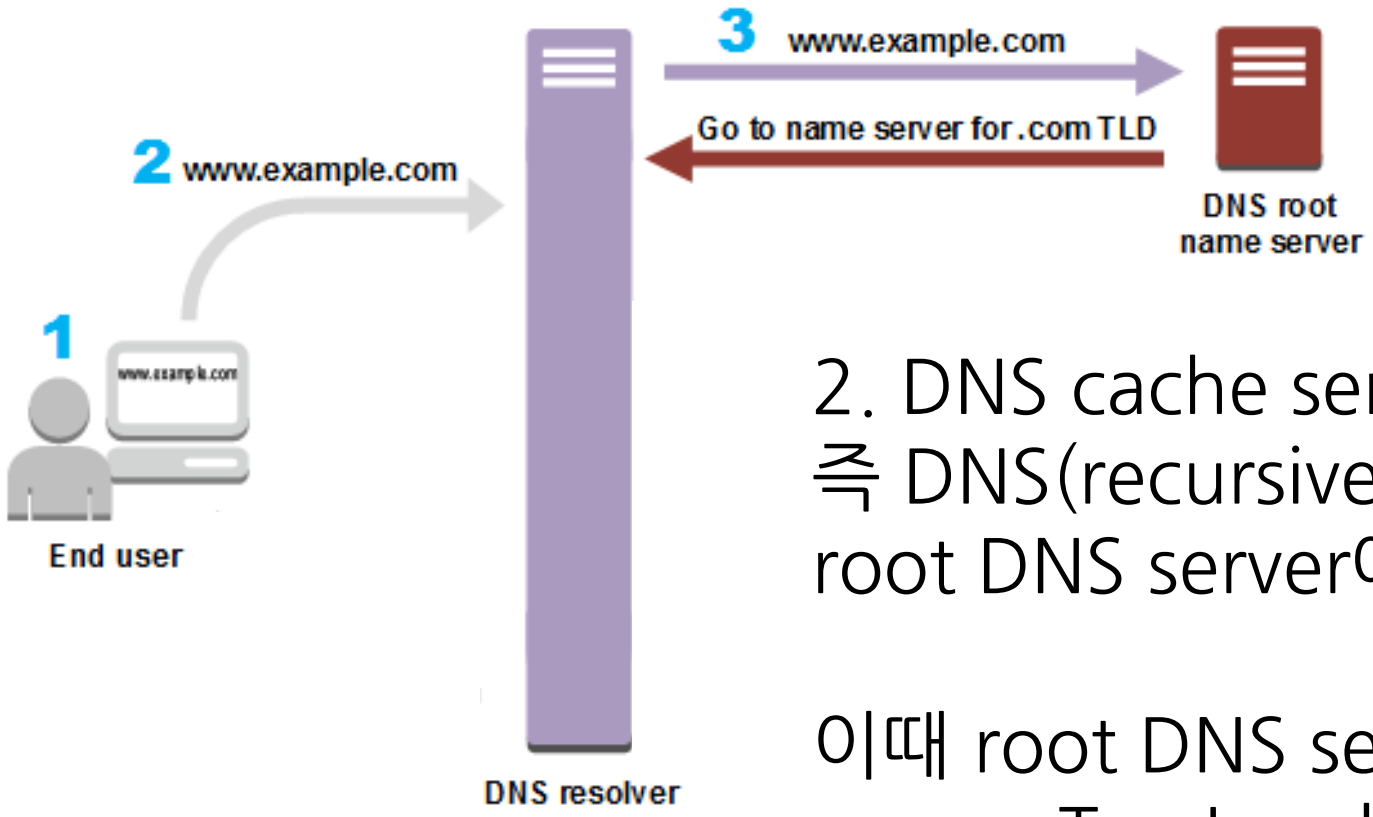
DNS서버의 작동 원리



1. DNS cache server,
즉 DNS(recursive)resolver가 요청을 받는다.

맵핑 ip가 있으면 그 ip를 리턴 한 뒤 요청을 보낸다.

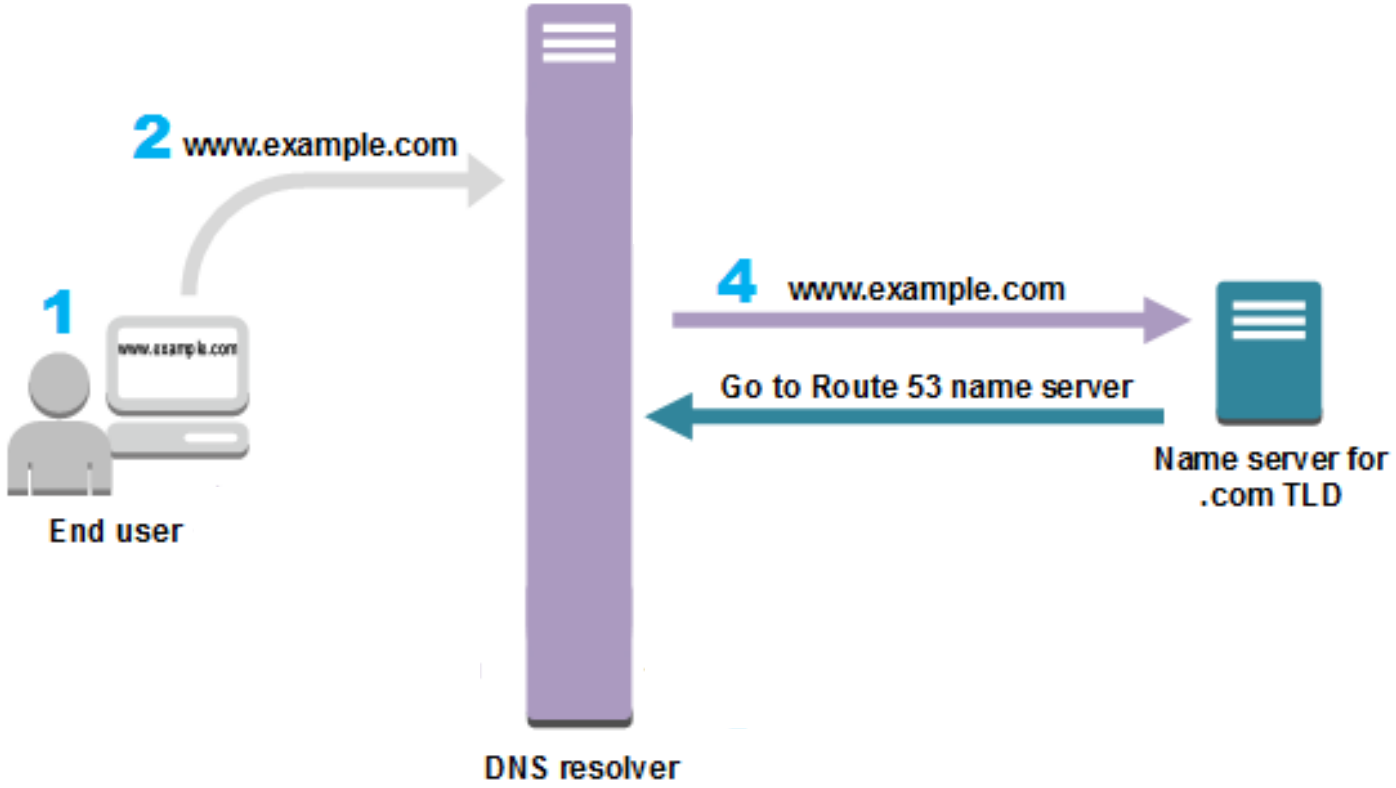
DNS서버의 작동 원리



2. DNS cache server,
즉 DNS(recursive)resolver에 맵핑 ip가 없으면 root DNS server에 요청을 보낸다.

이때 root DNS serve는 *.com을 다루는 DNS server, Top Level Domain Server list를 보낸다.

DNS서버의 작동 원리

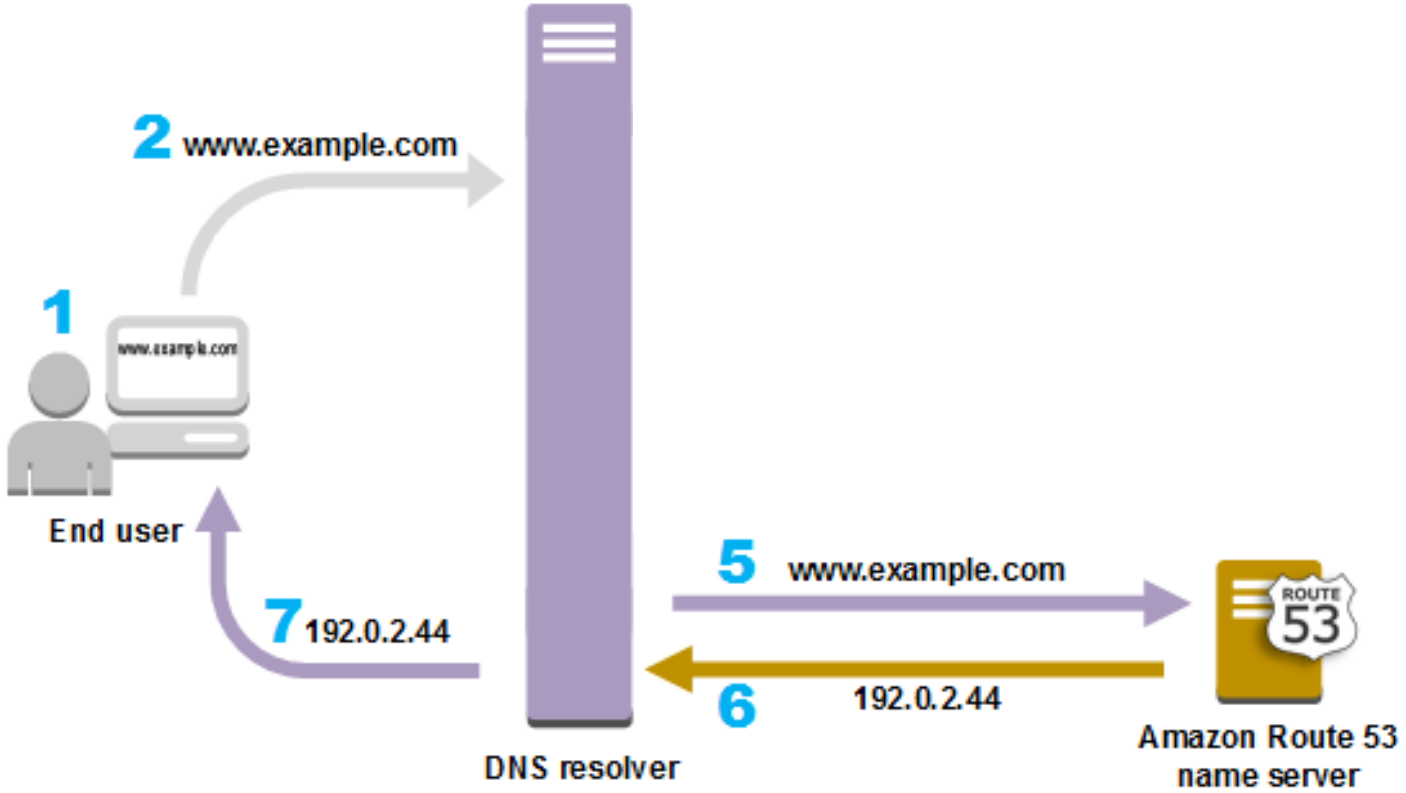


계속 요청을 반복해서
Recursive resolver라 한다.

3. *.com을 다루는 DNS server, Top Level Domain Server list를 순회하면서 example을 아냐고 물어보며,

각 TLD DNS server는
*.Example을 담당하는
DNS server,
즉 Authoritative name server list를 준다.

DNS서버의 작동 원리



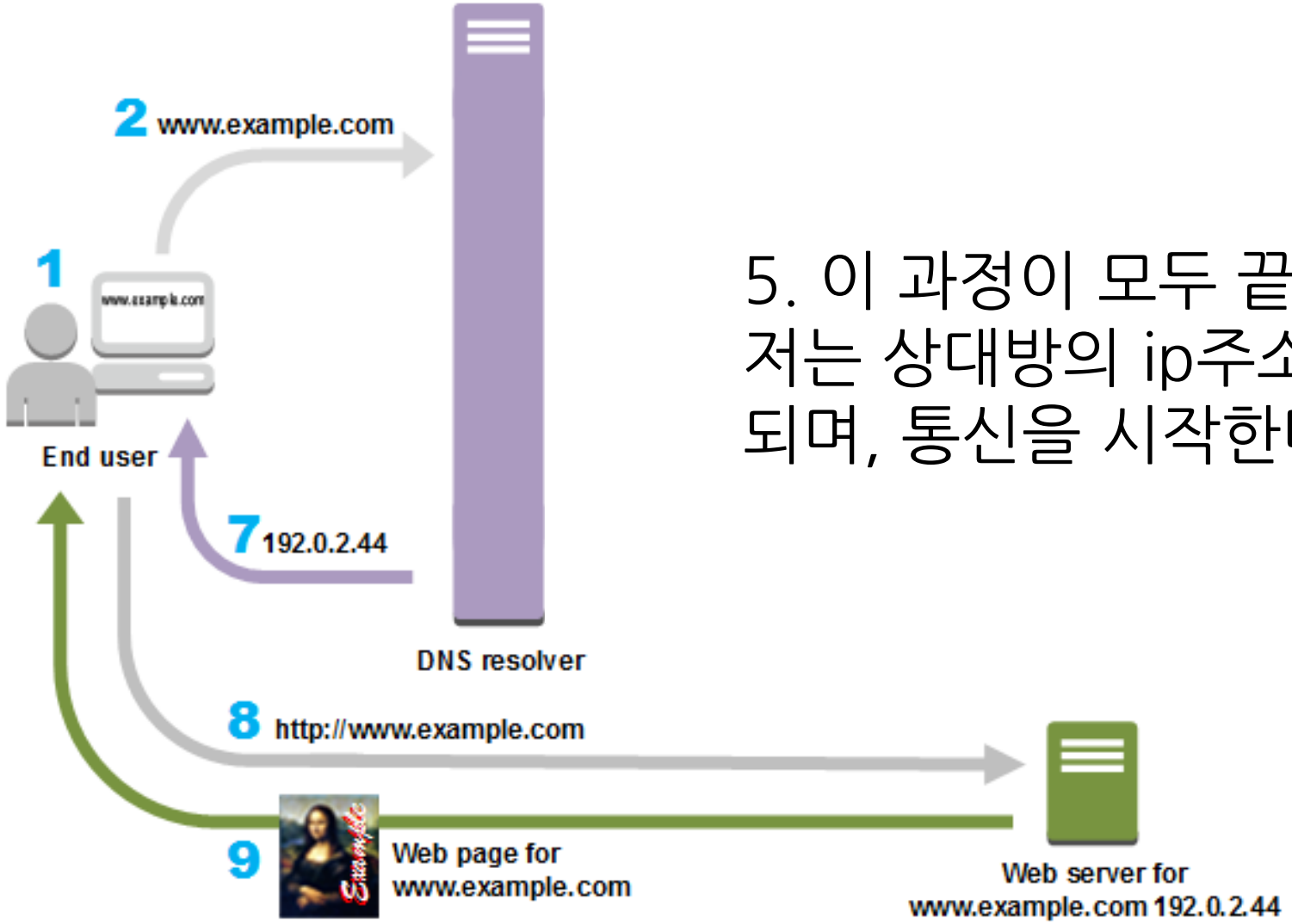
4. 이 example을 관리하는 Authoritative name server 에서 www을 아냐 물어보고,

이에 맞는 ip를 보내준다.

이제 이를 받아 요청자에게 보내준다.

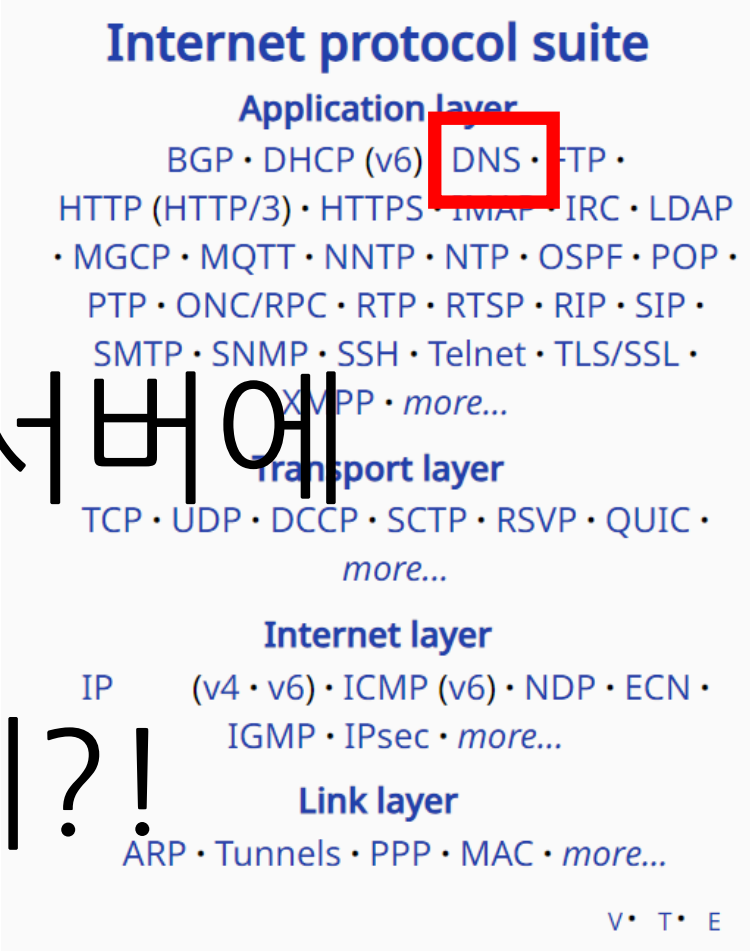
계속 요청을 반복해서
Recursive resolver라 한다.

DNS서버의 작동 원리



5. 이 과정이 모두 끝나야 유저는 상대방의 ip주소를 알게 되며, 통신을 시작한다.

DNS



그래서 어떻게 DNS서버에
요청을 보내서
url -> ip를 얻는데?!

= DNS프로토콜로 통신해서!

DNS 메시지 형식

헤더 + 데이터

익숙하군..

DNS 메시지 형식

출처 :
http://www.ktword.co.kr/test/view/view.php?no=2918

1. DNS 질의/응답의 메세지 및 동작 형태

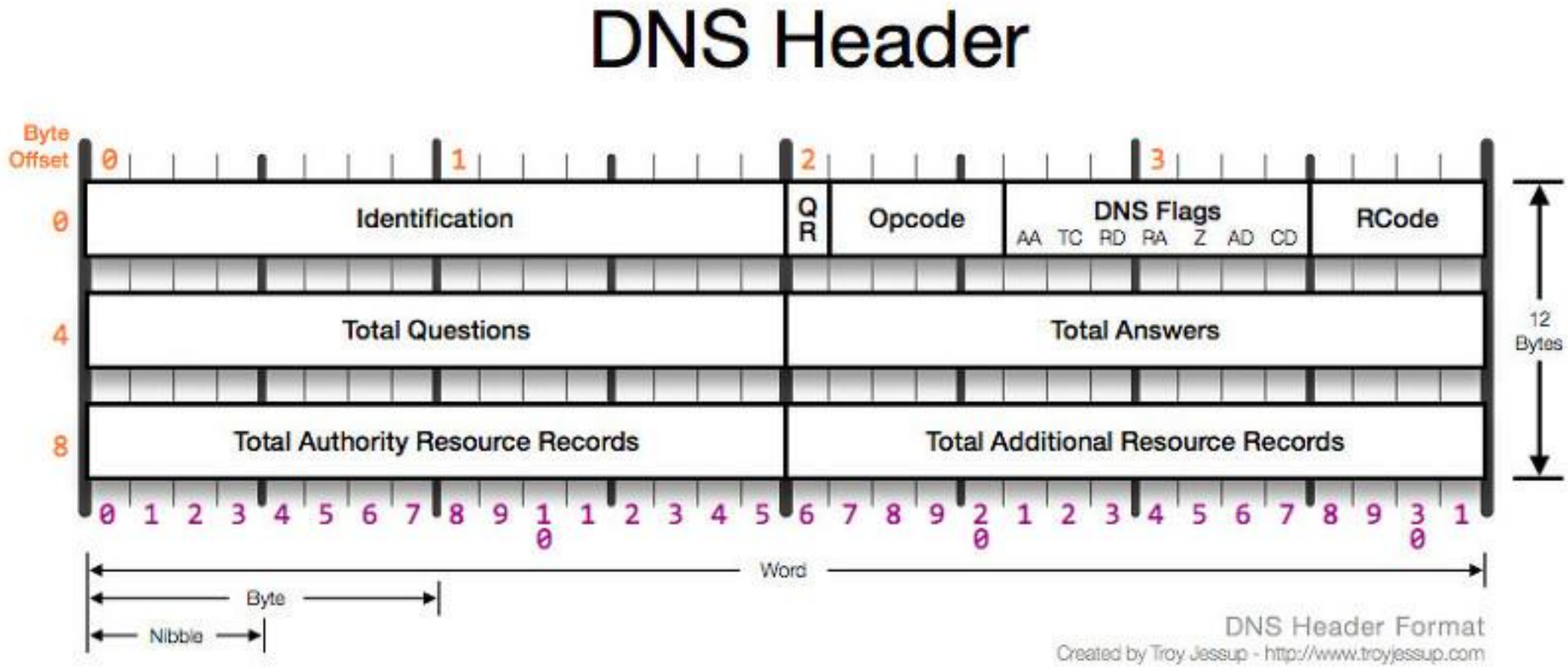
○ DNS 메세지 종류 ☞ DNS 헤더, DNS 메세지 참조

- DNS 질의 메세지(2개 영역 만 있음) = Header + 질의
- DNS 응답 메세지(5개 영역 있음) = Header + (질의 + 응답 + 책임 + 부가정보)

그냥 URL <-> IP를 할 때,
이 방식대로 DNS서버에 요청하면
이런 방식으로 응답 할게요~ 를 표현 한 것이다.

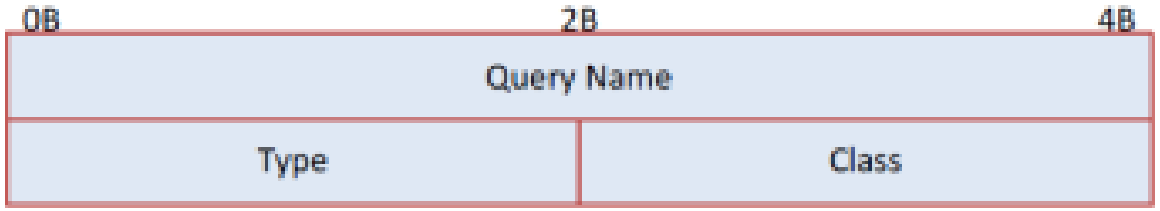
단어가 딱딱하다고 어지러워 하지 말자. 뒤의 그림으로 알아보자.

DNS **header** 형식 (그냥 이렇게 생겼구나~)

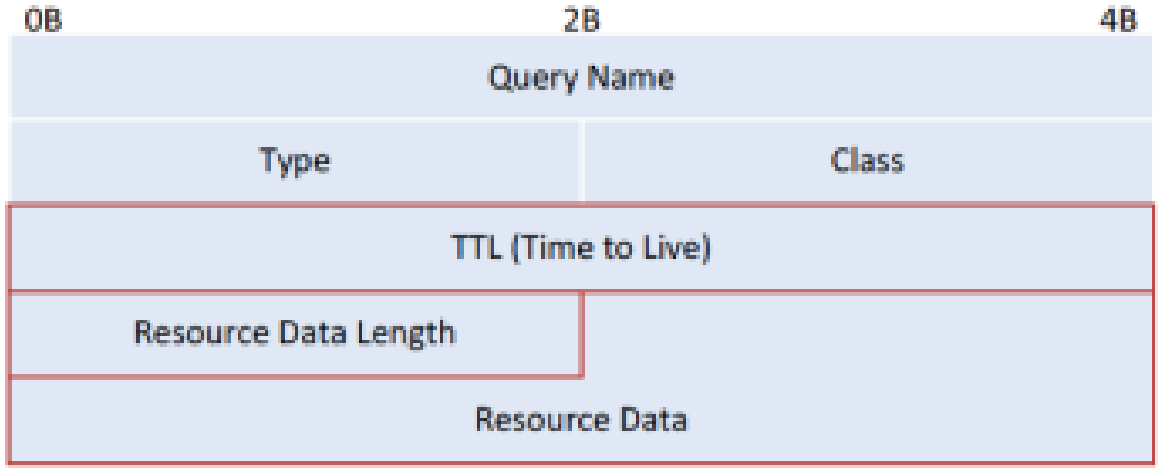


출처 : https://blog.naver.com/gaegurijump/110186379065

DNS **data** 형식 (그냥 이렇게 생겼구나~)



(a) DNS 질의 레코드



(b) DNS 응답 레코드

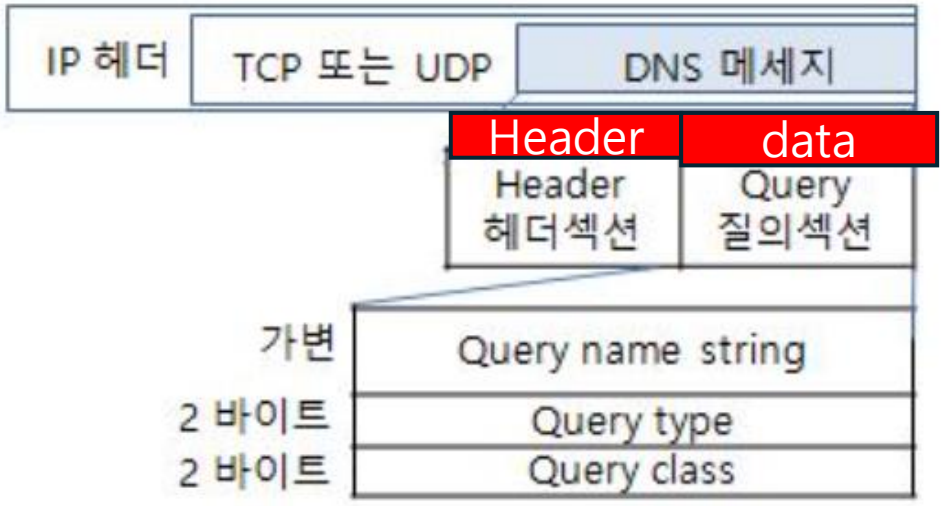
즉 url정보를 보내면
거기에 ip랑 TTL정보를 추가해서 다시 보내주는구나~

DNS **Request** message form

5. DNS 메세지 질의 영역 (DNS Query Section) : 가변 길이

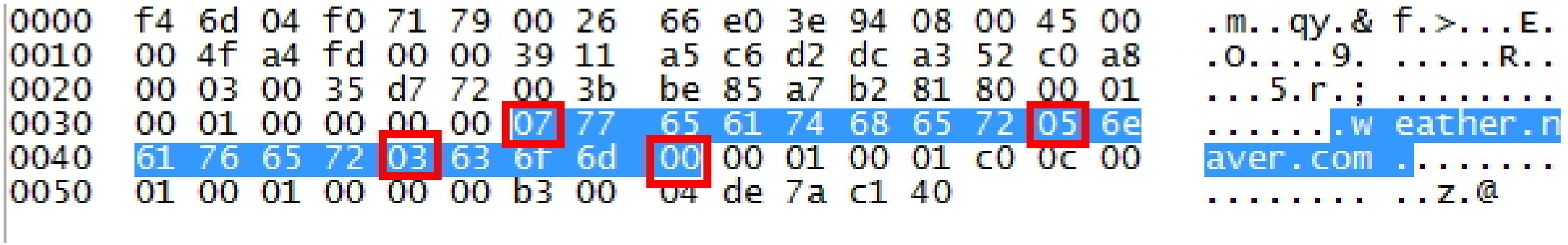
ㅇ 질의 영역은, 1 이상의 질의/문의 레코드들로 구성됨

- 질의 메세지,응답 메세지 모두에 존재할 수 있음
- 단, 영역 내 자원레코드 항목들은, DNS 응답 메세지에 만 있고, DNS 질의 메세지에는 없음



출처 :
http://www.ktword.co.kr/test/view/view.php?m_temp1=2194

DNS **Request** message form



실제 DNS 요청 패킷은 대충 이렇게 생겼다.
이 패킷을 프로토콜에 맞게 바꾸면 다음과 같이 이해 할 수 있다.

출처 : https://darksoulstory.tistory.com/62

DNS **Request** message form

```
0000   f4 6d 04 f0 71 79 00 26  66 e0 3e 94 08 00 45 00   .m..qy.& f.>...E.
0010   00 4f a4 fd 00 00 39 11  a5 c6 d2 dc a3 52 c0 a8   .O....9. .....R..
0020   00 03 00 35 d7 72 00 3b  be 85 a7 b2 81 80 00 01   ...5.r.; ........
0030   00 01 00 00 00 00 07 77  65 61 74 68 65 72 05 6e   .......w eather.n
0040   61 76 65 72 03 63 6f 6d  00 00 01 00 01 c0 0c 00   aver.com .........
0050   01 00 01 00 00 00 b3 00  04 de 7a c1 40            ........ ..z.@
```

Message = header + data

Header

data

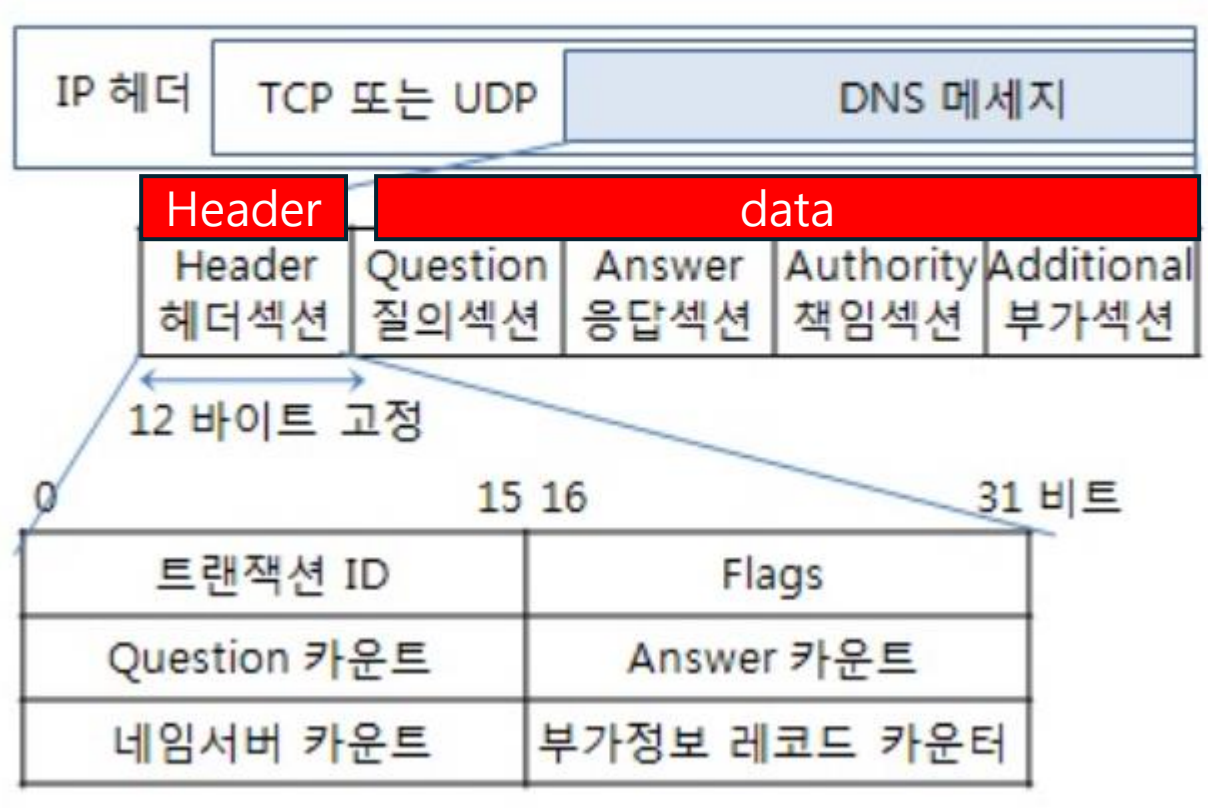
```
Domain Name System (query)
  [Response In: 11162]
  Transaction ID: 0x4caa
  Flags: 0x0100 (Standard query)
    0... .... .... .... = Response: Message is a query
    .000 0... .... .... = Opcode: Standard query (0)
    .... ..0. .... .... = Truncated: Message is not truncated
    .... ...1 .... .... = Recursion desired: Do query recursively
    .... .... .0.. .... = Z: reserved (0)
    .... .... ...0 .... = Non-authenticated data: Unacceptable
  Questions: 1
  Answer RRs: 0
  Authority RRs: 0
  Additional RRs: 0
  Queries
    search.naver.com: type A, class IN
      Name: search.naver.com
      Type: A (Host address)
      Class: IN (0x0001)
```

DNS

그럼 최종 response는?

DNS **Response** message form

1. DNS 메세지 헤더 영역 (필수, 12 바이트 고정)



출처 :
http://www.ktword.co.kr/test/view/view.php?m_temp1=2194

출처 : https://youngq.tistory.com/56

DNS **Response** message form

Header

data

```
Domain Name System (response)
    [Request In: 11161]
    [Time: 0.012197000 seconds]
    Transaction ID: 0x4caa
  Flags: 0x8180 (Standard query response, No error)
      1... .... .... .... = Response: Message is a response
      .000 0... .... .... = Opcode: Standard query (0)
      .... .0.. .... .... = Authoritative: Server is not an authority for domain
      .... ..0. .... .... = Truncated: Message is not truncated
      .... ...1 .... .... = Recursion desired: Do query recursively
      .... .... 1... .... = Recursion available: Server can do recursive queries
      .... .... .0.. .... = Z: reserved (0)
      .... .... ..0. .... = Answer authenticated: Answer/authority portion was not authenticated by the server
      .... .... ...0 .... = Non-authenticated data: Unacceptable
      .... .... .... 0000 = Reply code: No error (0)
    Questions: 1
    Answer RRs: 1
    Authority RRs: 0
    Additional RRs: 0
  Queries
    search.naver.com: type A, class IN
        Name: search.naver.com
        Type: A (Host address)
        Class: IN (0x0001)
  Answers
    search.naver.com: type A, class IN, addr 114.111.42.141
        Name: search.naver.com
        Type: A (Host address)
        Class: IN (0x0001)
        Time to live: 3 minutes, 15 seconds
        Data length: 4
        Addr: 114.111.42.141 (114.111.42.141)
```

Message = header + data

Request에서 보낸 Query에
Answers 항목이 붙어서 옴

여기에 ip, ttl이 있다.

DNS의 취약점

1. 우리 집 컴퓨터가 해킹을 당했다. 해커가 hosts나 DNS cache에서 www.google.com을 다른 ip로 맵핑 하도록 바꾸면..?

2. DNS서버가 해킹 당했다. 해커가 DNS mapping을 삭제하거나 바꿔버리면…?

3. 해커가 13개의 root DNS server를 모두 없애 버리면..?