

# Introduction to Computer Security

## Project 1: DNS Reflection and Amplification Attacks

Chi-Yu Li (2019 Spring)  
Computer Science Department  
National Chiao Tung University

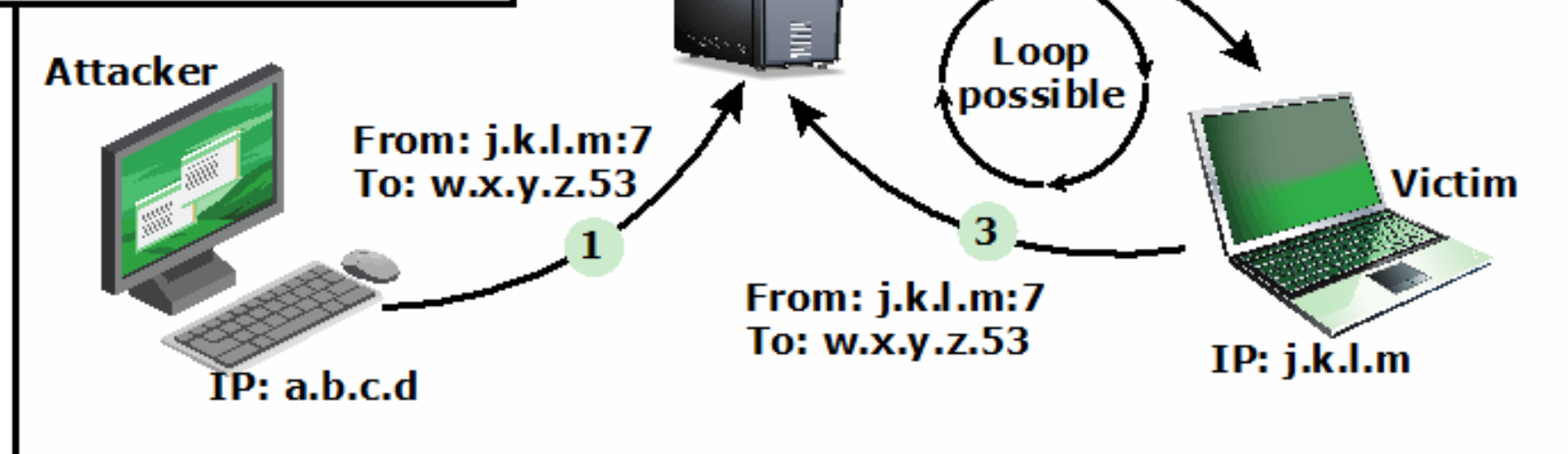
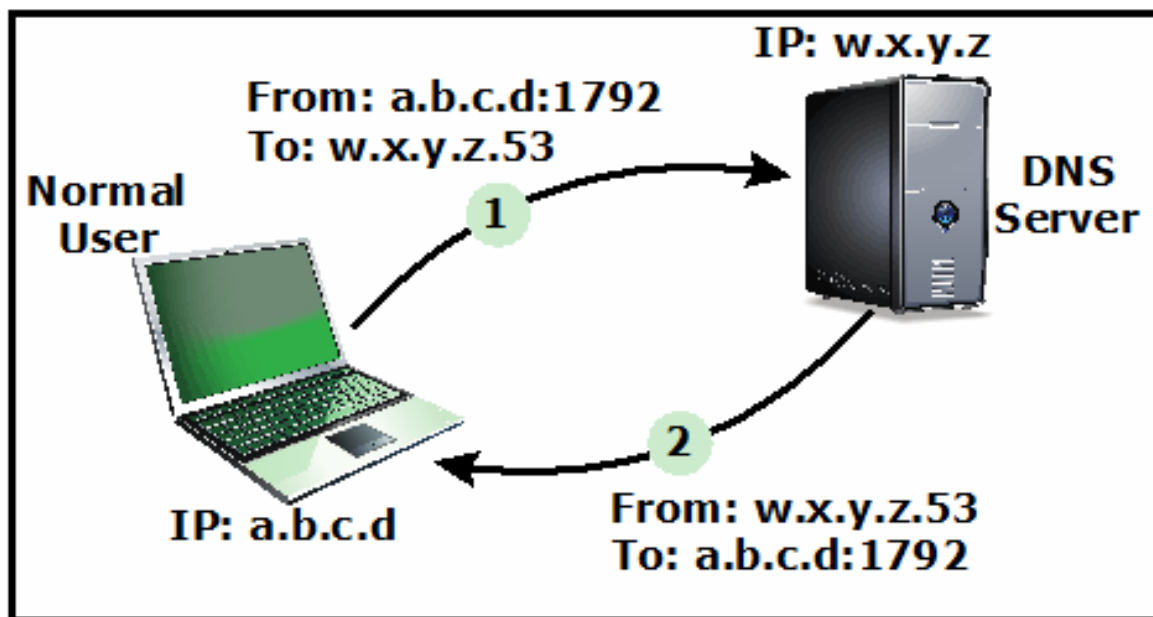
# Goals

- Understand how to launch DNS reflection and amplification attacks and then defend against them
- You will learn how to
  - Program with raw sockets
  - Generate IP packets with spoofed IP addresses
  - Trace packets using Wireshark
  - <sup>捏造</sup>Fabricate DNS query messages
  - Launch DNS reflection and amplification attacks

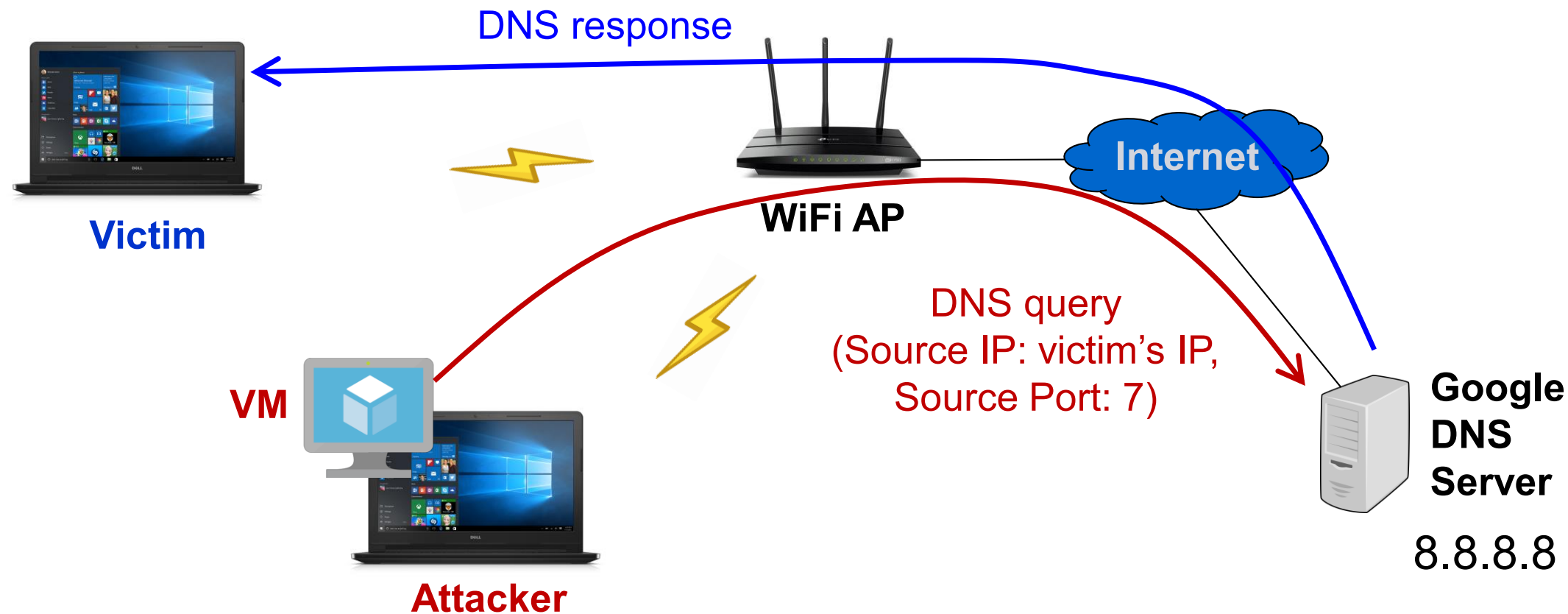
# Requirements

- You need to develop/run your program in a given virtual machine
  - VMware Workstation Player: Please download it from [VMware](#)
  - VM image: Please download it from [Link](#)
    - Username/password: cs2019/cs2019
- The language you use must be C/C++
- You are allowed to team up. Each team has at most 2 students
  - Teams: discussions are allowed, but no collaboration
- Please submit your source codes and then use them to show your demo
- TA: [Yu-Yen Huang](#) (hyyisgood@gmail.com)

# What is the DNS Reflection Attack?



# Your DNS Reflection Attack



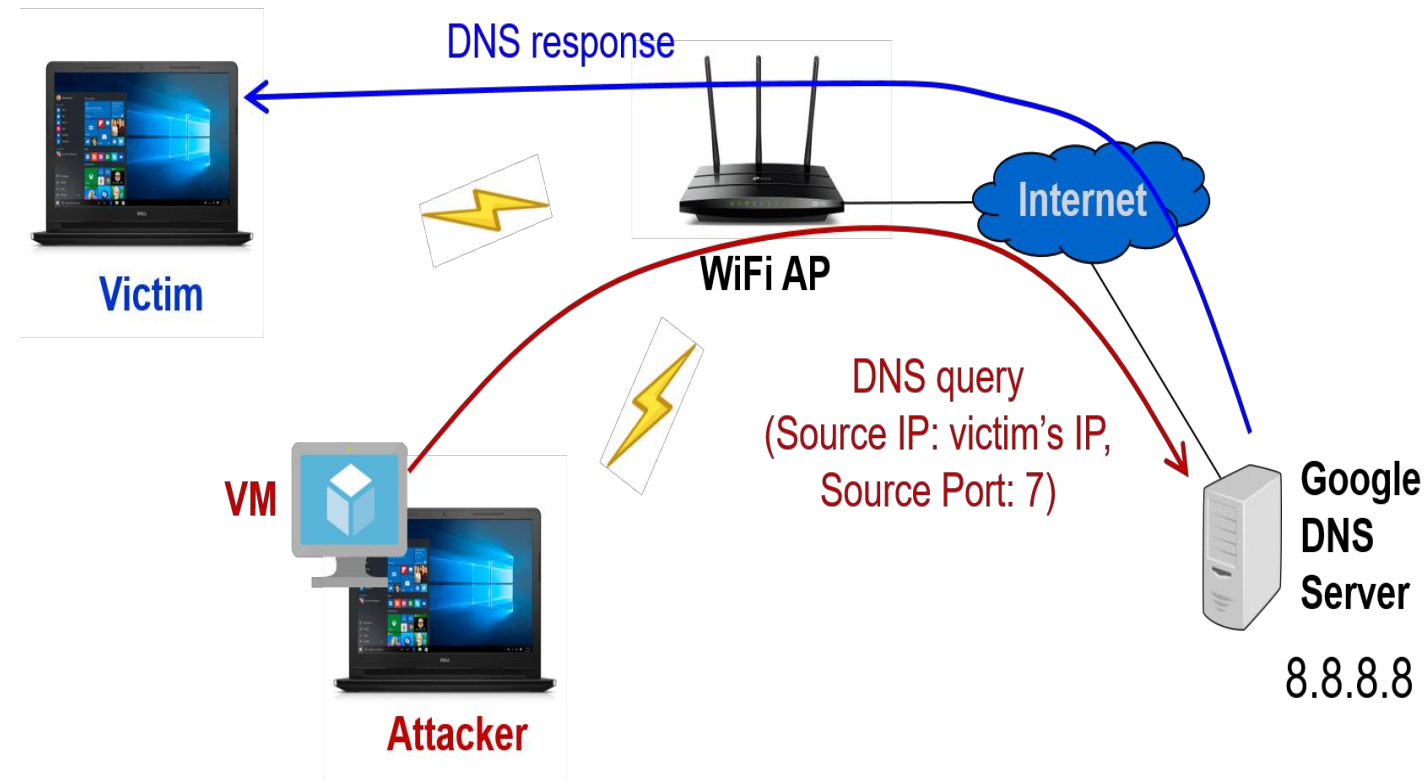
# Two Tasks

- Task I: DNS reflection attack (50%)
- Task II: DNS amplification attack (35%)
  - Amplification ratio:  $R = S_r/S_q$ 
    - $S_q$ : packet size of the DNS query
    - $S_r$ : packet size of the DNS response
  - $3 \leq R < 6$ : **20%**,  $6 \leq R < 10$ : **25%**,  $10 \leq R$ : **35%**
- Demo Q&A (15%)

# Task I: DNS Reflection Attack

(Given a DNS server's IP and the victim's IP)

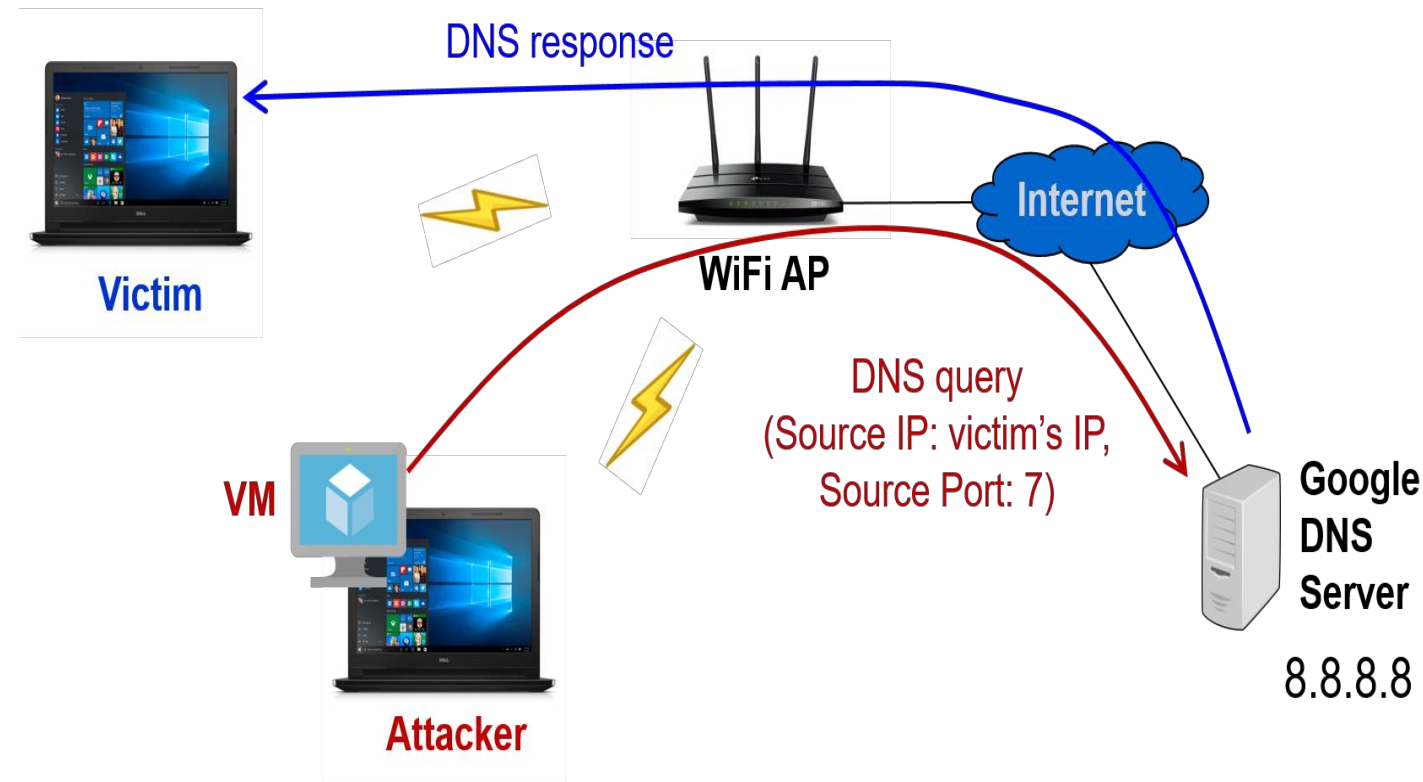
- (Attacker) **捏造** Fabricates a DNS query message with a UDP packet
- (Victim) Uses Wireshark to check whether the victim receives the DNS response



# Task II: DNS Amplification Attack

(Given a DNS server's IP and the victim's IP)

- (Attacker) Fabricates a DNS query message with a UDP packet
  - ❑ Checks its packet size using Wireshark
- (Victim) Uses Wireshark to check the packet size of the DNS response
  - ❑ Obtains the amplification ratio

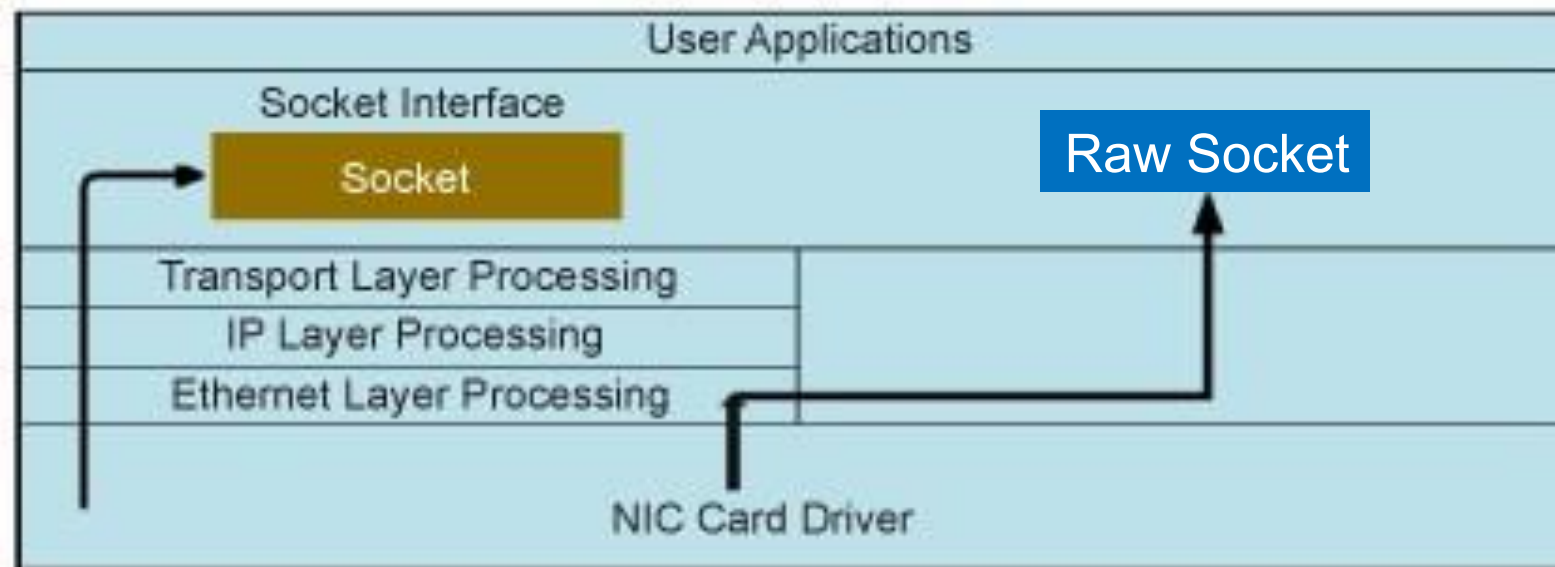




# Hint I: How to Create IP <sup>欺騙</sup> Spoofing Packets?

- Using Raw Socket

- Normal network sockets vs. Raw sockets



Source of figures: A Guide to Using Raw Sockets by Subodh Saxena

# Hint I: How to Create IP Spoofing Packets? (Cont.)

- Implementation using Raw sockets

- Create a raw socket with UDP protocol

```
sd = socket(PF_INET, SOCK_RAW, IPPROTO_UDP)
```

- Fabricate the IP header

```
struct ipheader *ip = (struct ipheader *) buffer;  
ip->iph_ihl = 5;
```

```
....
```

```
ip->iph_souceip = inet_addr(argv[1]);
```

```
....
```

- Fabricate the UDP header

```
struct udphheader *udp = ...  
udp->udph_srcport = htons(atoi(argv[2]));
```

```
....
```

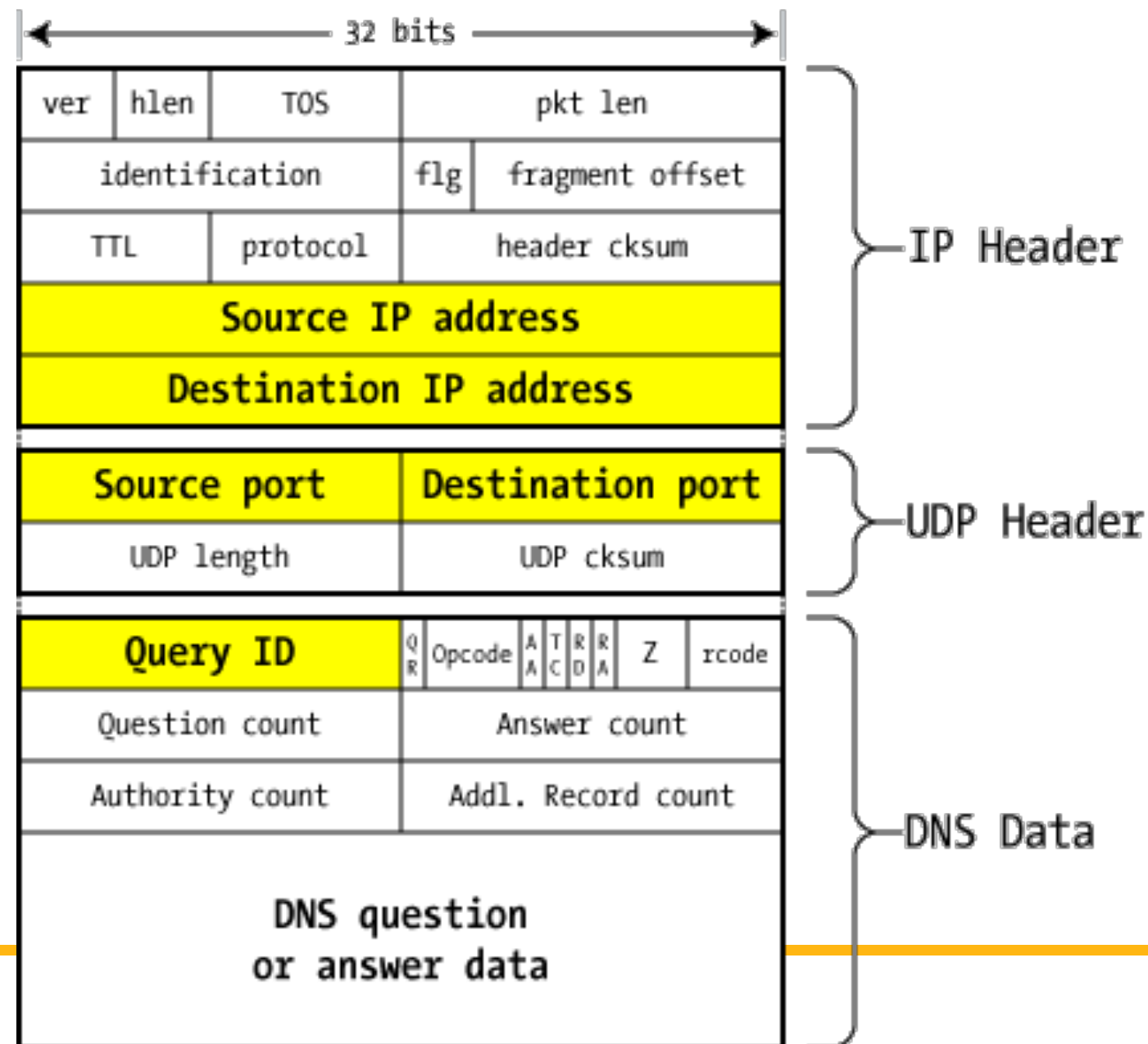
- Calculate the checksum over IP and UDP headers

- Create DNS query in the UDP payload

- Reference: [Tutorial](#) [Sample code](#)

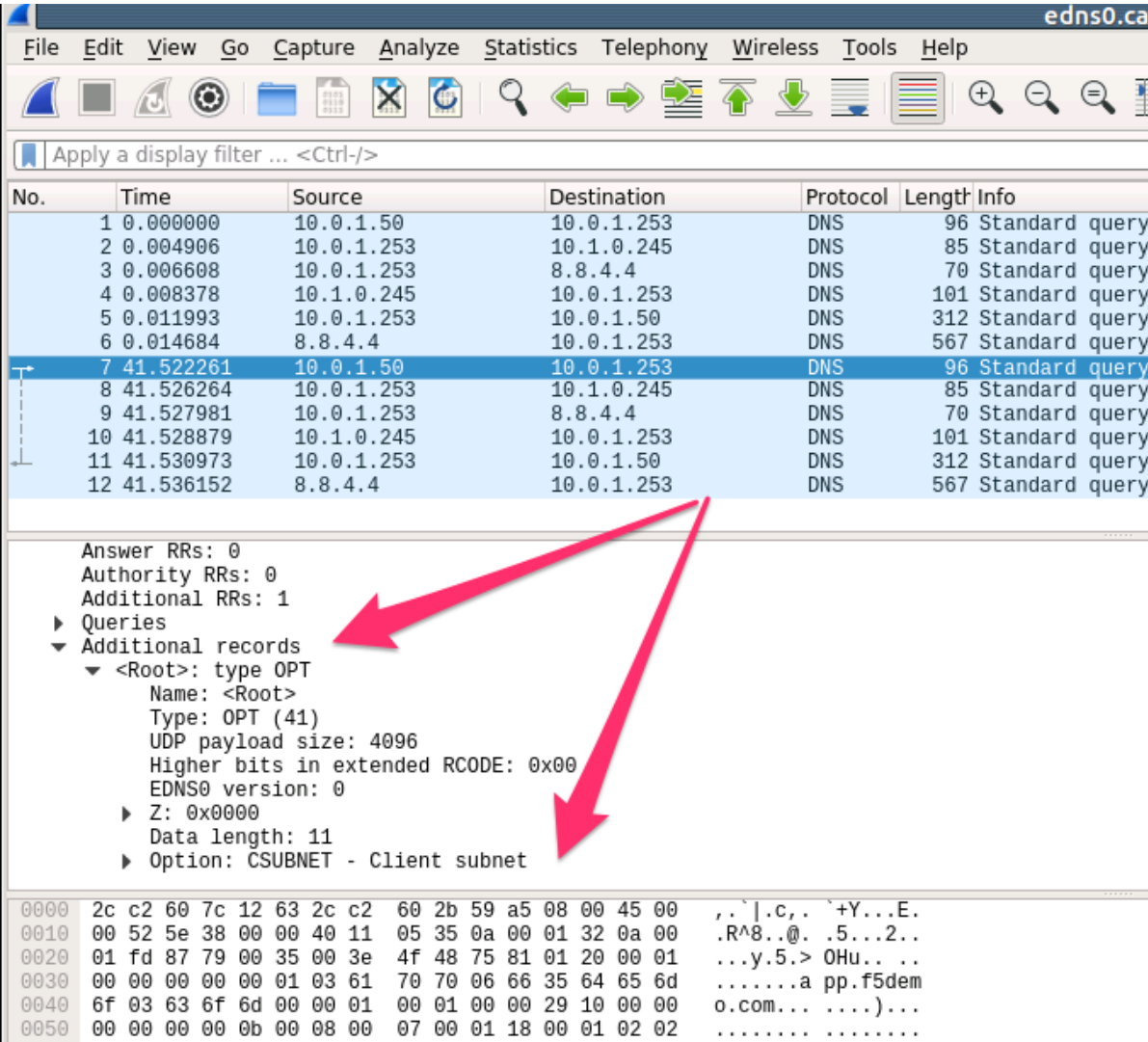
# Hint I: How to Create IP Spoofing Packets? (Cont.)

- DNS/UDP/IP packet format



## Hint 2: How to Create DNS Query Message?

- Generate a DNS query (e.g., using ping) and then capture it using Wireshark
- Fill in the content based on the observation from Wireshark



The image shows a Wireshark capture of DNS traffic. The packet list at the top shows several DNS queries. Packet 7 is selected, showing a standard query from 10.0.1.50 to 10.0.1.253. The packet details pane below shows the structure of the DNS message:

- Answer RRs: 0
- Authority RRs: 0
- Additional RRs: 1
- Queries
- Additional records
  - <Root>: type OPT
    - Name: <Root>
    - Type: OPT (41)
    - UDP payload size: 4096
    - Higher bits in extended RCODE: 0x00
    - EDNS0 version: 0
    - Z: 0x0000
    - Data length: 11
    - Option: CSUBNET - Client subnet

Two red arrows point from the packet list to the details pane: one from packet 7 and another from packet 12, which is also a standard query from 10.0.1.50 to 10.0.1.253.

The packet bytes pane at the bottom shows the raw data in hexadecimal and ASCII:

```

0000  2c c2 60 7c 12 63 2c c2 60 2b 59 a5 08 00 45 00  ,. |.c,. +Y...E.
0010  00 52 5e 38 00 00 40 11 05 35 0a 00 01 32 0a 00  .R^8..@. .5...2..
0020  01 fd 87 79 00 35 00 3e 4f 48 75 81 01 20 00 01  ...y.5.> 0Hu.. ..
0030  00 00 00 00 00 01 03 61 70 70 06 66 35 64 65 6d  .....a pp.f5dem
0040  6f 03 63 6f 6d 00 00 01 00 01 00 00 29 10 00 00  o.com... ..)....
0050  00 00 00 00 0b 00 08 00 07 00 01 18 00 01 02 02  .....
  
```

# Project Submission

- Due date: 3/27 11:55pm
- Submission rules
  - ❑ Put all your files into a directory, the name of which is your student ID, and upload its zip file to New e3
    - Don't include executable files
  - ❑ If your team has two members, please create a blank file with the name as the concatenation of your IDs separated by “-”
  - ❑ Sample: 0878778.zip
    - 0878778-0889008
    - Makefile
    - dns\_attack.cpp
    - ....

# Demo

- Date: 3/28 (9:30am-5pm) @ EC315
- TA will prepare your zip file for you to demo
- You will
  - ❑ be asked to launch the amplification attack from one PC, and verify whether it succeeds and its ratio using Wireshark at another PC
  - ❑ be only allowed to “make” and input required commands
    - A Makefile is needed to compile your source files
  - ❑ be not allowed to modify your codes
  - ❑ be asked some questions (15%)
    - e.g., how do you do DNS amplification attack (explanation with your codes)? How to defend against this attack?
- ❑ be responsible to show the traces to TA and explain why you have successfully launched attacks and the amplification ratio

# Questions?